# Path Detection:
# A Quantum Computing Primitive

**Shelby Kimmel**

**Middlebury College**

Based on work with
Stacey Jeffery: arXiv: 1704.00765 (Quantum vol 1 p 26)
Michael Jarret, Stacey Jeffery, Alvaro Piedrafita, *in progress*

Middlebury

# How to make quantum algorithms accessible?

# How to make quantum algorithms accessible?

- Need quantum algorithmic primitives

# How to make quantum algorithms accessible?

- Need quantum algorithmic primitives
  1. Apply to a wide range of problems
  2. Easy to understand and analyze (without knowing quantum mechanics)

# How to make quantum algorithms accessible?

- Need quantum algorithmic primitives
  1. Apply to a wide range of problems
  2. Easy to understand and analyze (without knowing quantum mechanics)

  – Ex: Searching unordered list of $n$ items
    – Classically, takes $\Omega(n)$ time
    – Quantumly, takes $O(\sqrt{n})$ time

# How to make quantum algorithms accessible?

- Need quantum algorithmic primitives
    1. Apply to a wide range of problems
    2. Easy to understand and analyze (without knowing quantum mechanics)

    - Ex: Searching unordered list of $n$ items
        - Classically, takes $\Omega(n)$ time
        - Quantumly, takes $O(\sqrt{n})$ time

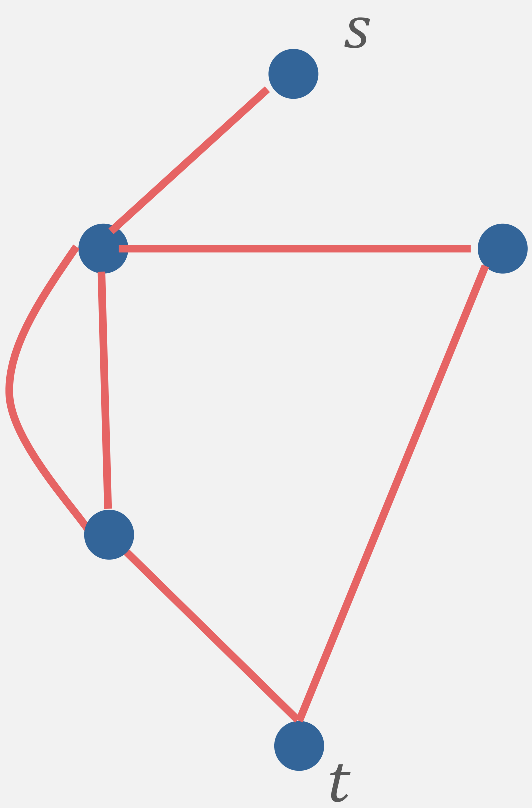
- New primitive: ***st*-connectivity**

# Outline:

A. Introduction to st-connectivity
B. st-connectivity makes a good algorithmic primitive
    1. Applies to a wide range of problems
    2. Easy to understand (without knowing quantum mechanics)
C. Applications and performance of algorithm

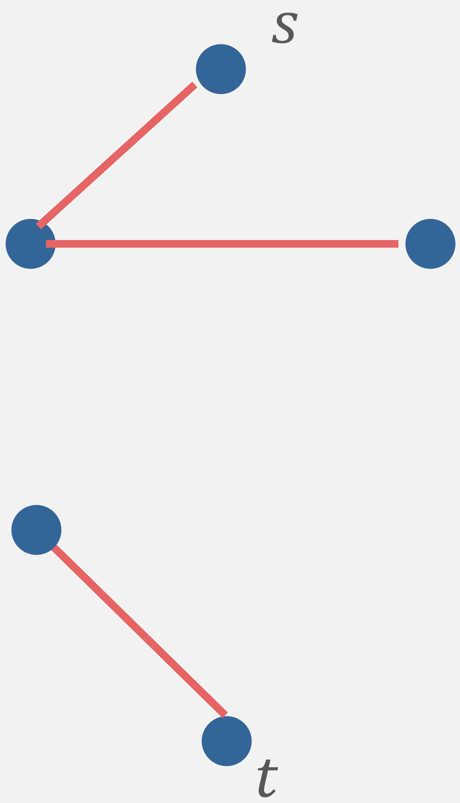
# $st$-connectivity

$\boldsymbol{st-connectivity}$:
is there a path from $s$ to $t$?

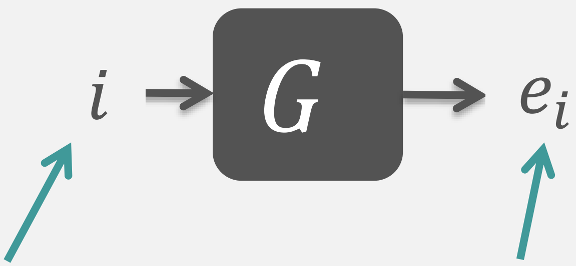


# $st$-connectivity

$\boldsymbol{st - connectivity}$:
is there a path from $s$ to $t$?

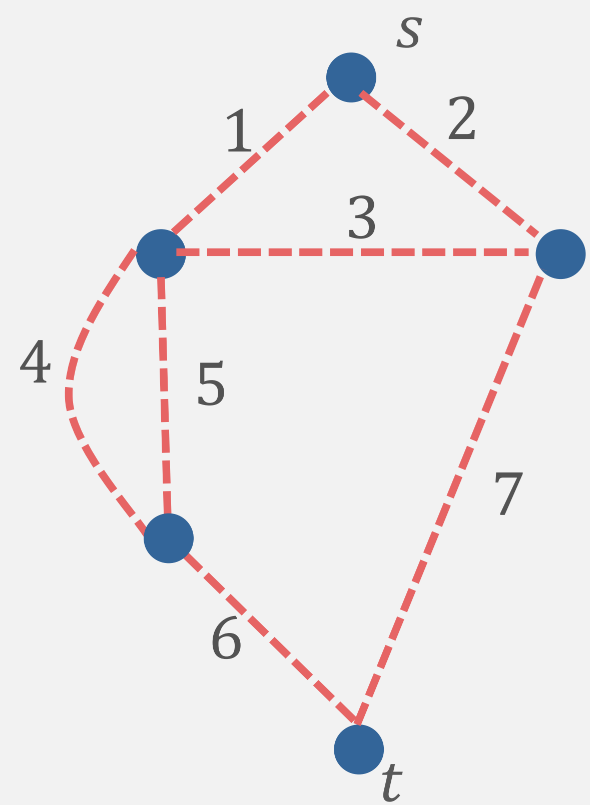


# Black Box Model




Edge label

- $e_i = 1$ if $i^{th}$ edge is there
- $e_i = 0$ if edge is not there




Let $\mathcal{H}$ be the set of graphs $G$ that the black box might contain.

# Figure of Merit

- Query Complexity
  - Number of uses (queries) of the black box
  - All other operations are free

- Under mild assumption, for our algorithm,

  quantum query complexity $\cong$ quantum time complexity

# Outline:

A. Introduction to st-connectivity
B. st-connectivity makes a good algorithmic primitive
   1. Applies to a wide range of problems

   2. Easy to understand (without knowing quantum mechanics)
C. Applications and performance of algorithm

# Outline:

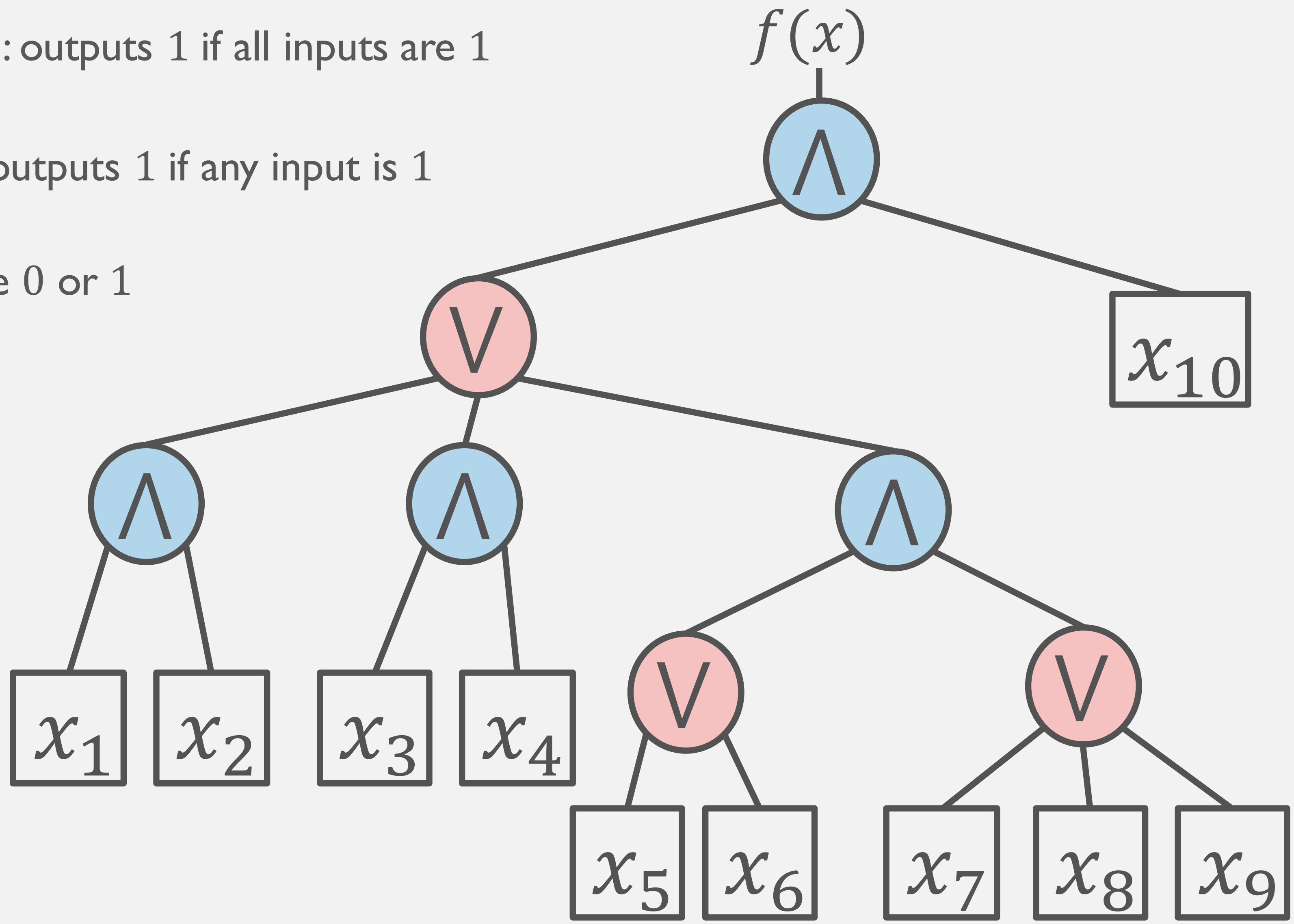
A. Introduction to st-connectivity
B. st-connectivity makes a good algorithmic primitive
   1. Applies to a wide range of problems
      - Evaluating Boolean formulas reduces to st-connectivity
   2. Easy to understand (without knowing quantum mechanics)
C. Applications and performance of algorithm

# Boolean Formulas

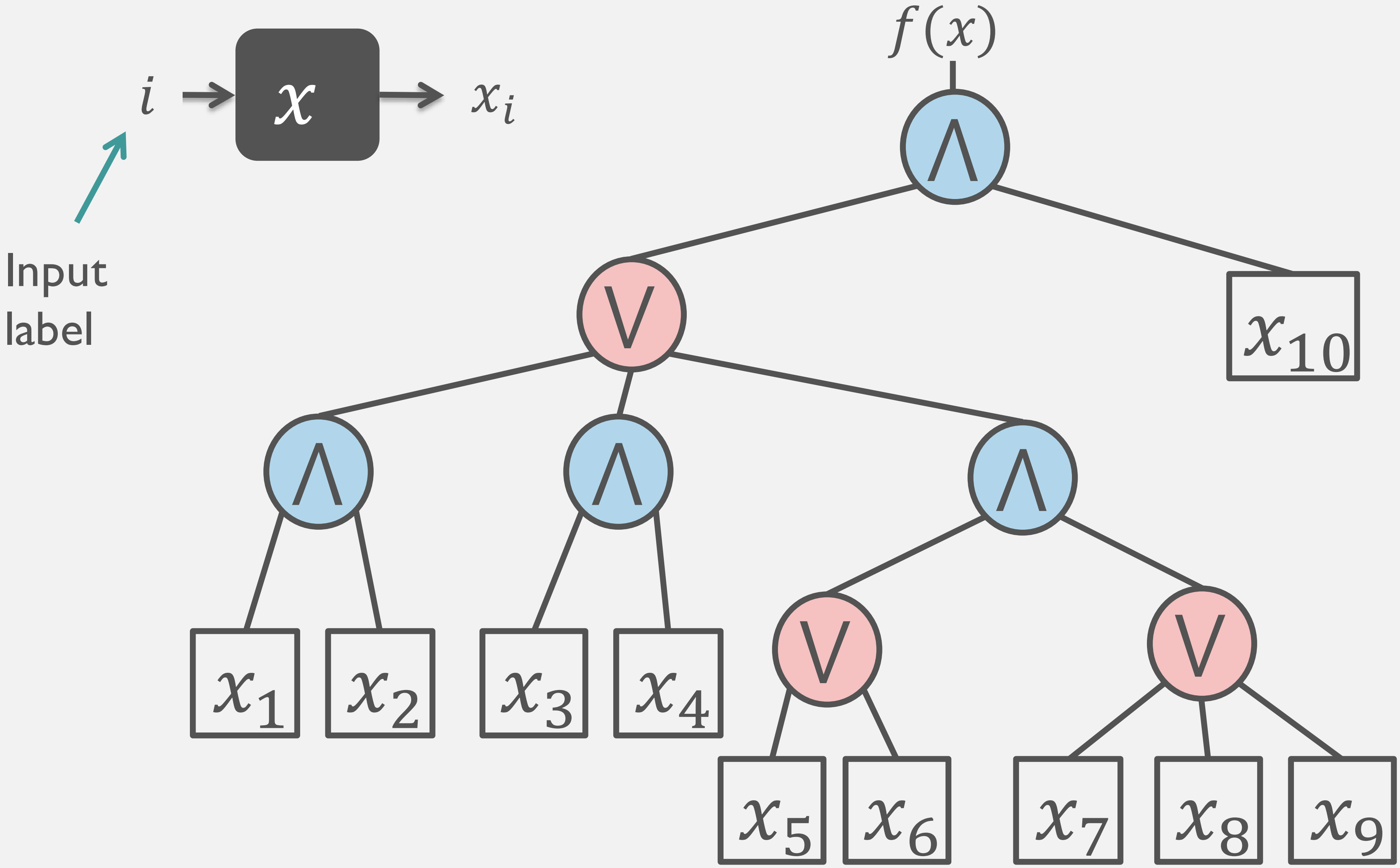
∧ $AND$: outputs 1 if all inputs are 1

∨ $OR$: outputs 1 if any input is 1

$x_i$ Value 0 or 1

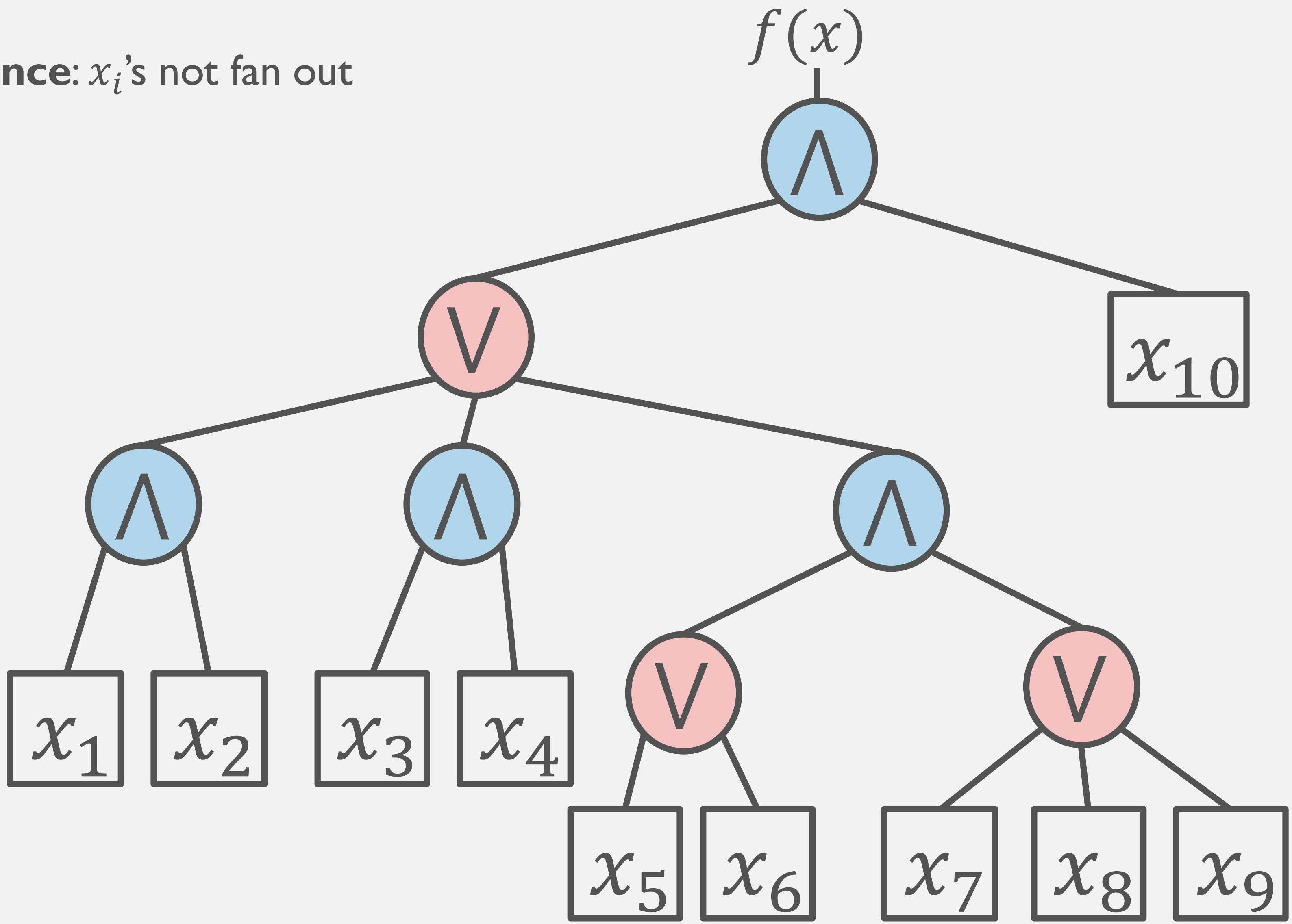


# Boolean Formulas



# Boolean Formulas

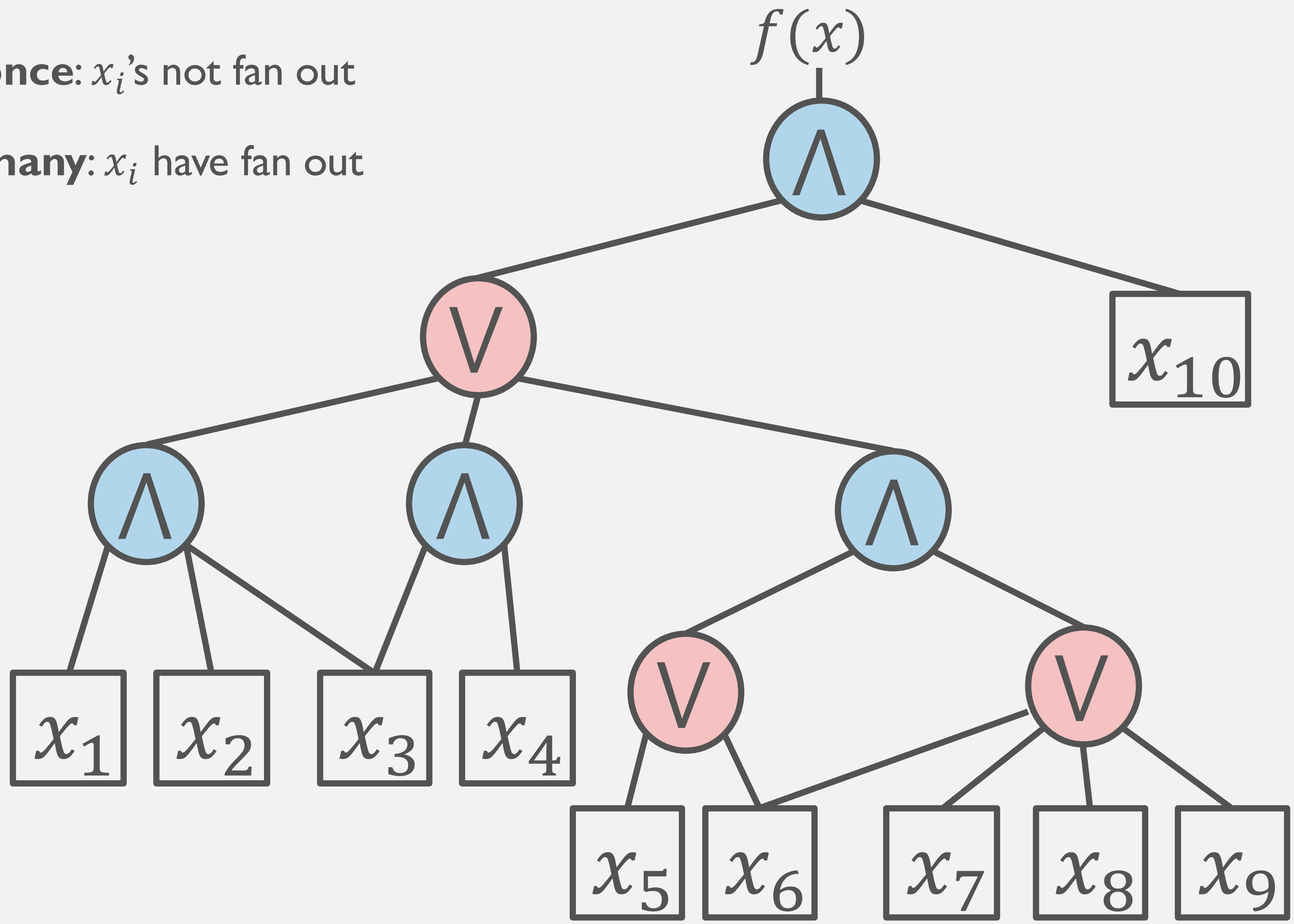
**Read-once**: $x_i$'s not fan out



# Boolean Formulas

**Read-once**: $x_i$'s not fan out

**Read-many**: $x_i$ have fan out

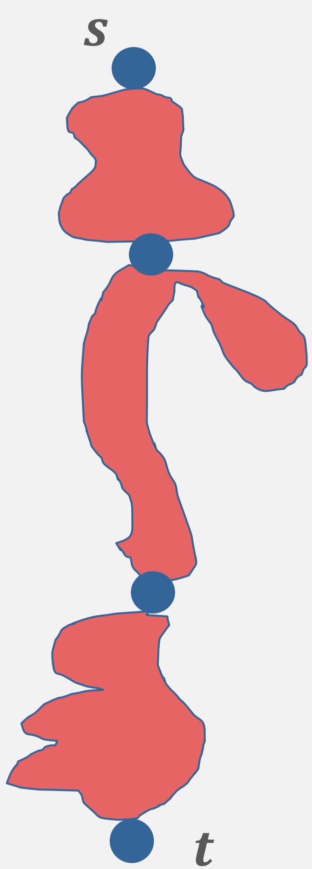


# Boolean Formula Applications

- Logic
- Designing electrical circuits
- Game theory (deciding who will win a game)
- Combinatorics and graph problems
- Linear programming
- Testing potential solution to an NP-complete problem

# Application to Boolean Formulas

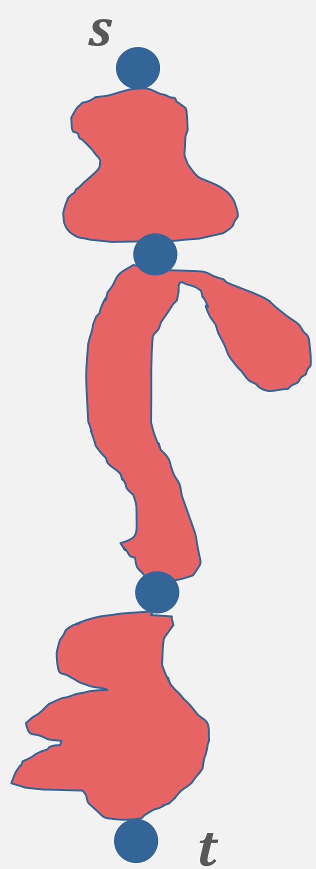
$\wedge$ $AND$: outputs 1 if all input subformulas have value 1




$s$ and $t$ are connected if all subgraphs are connected

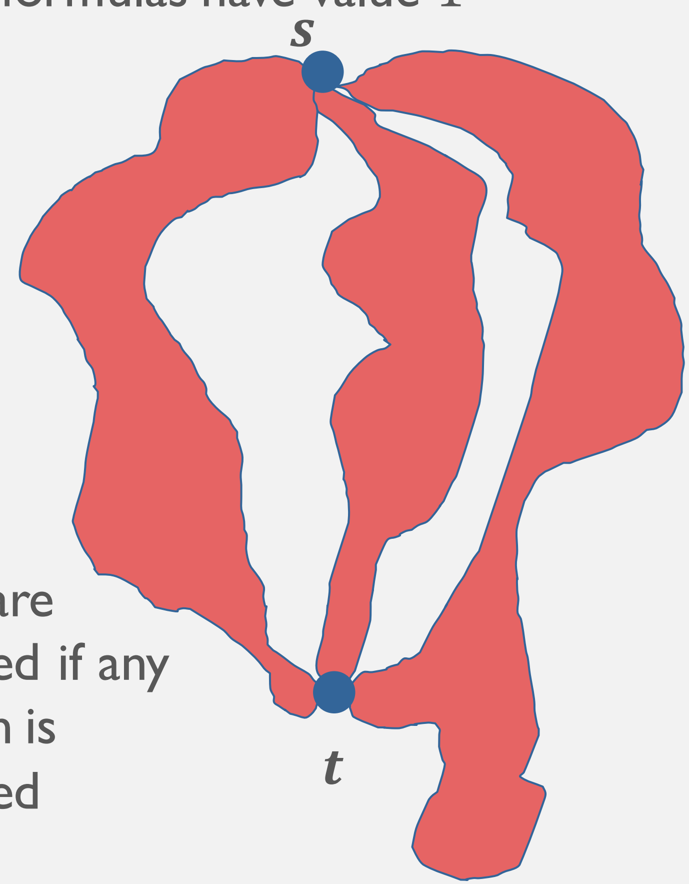
# Application to Boolean Formulas

$\wedge$ $AND$: outputs 1 if all input subformulas have value 1



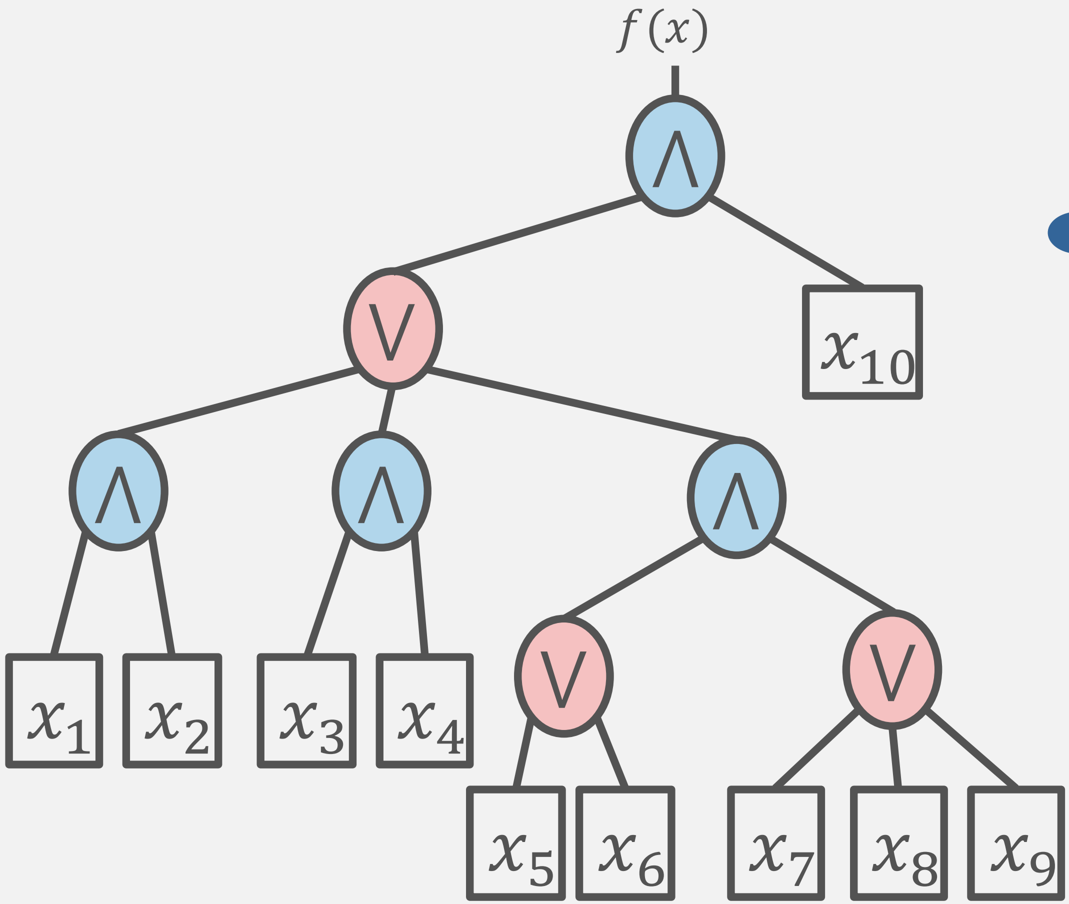

$s$ and $t$ are connected if all subgraphs are connected

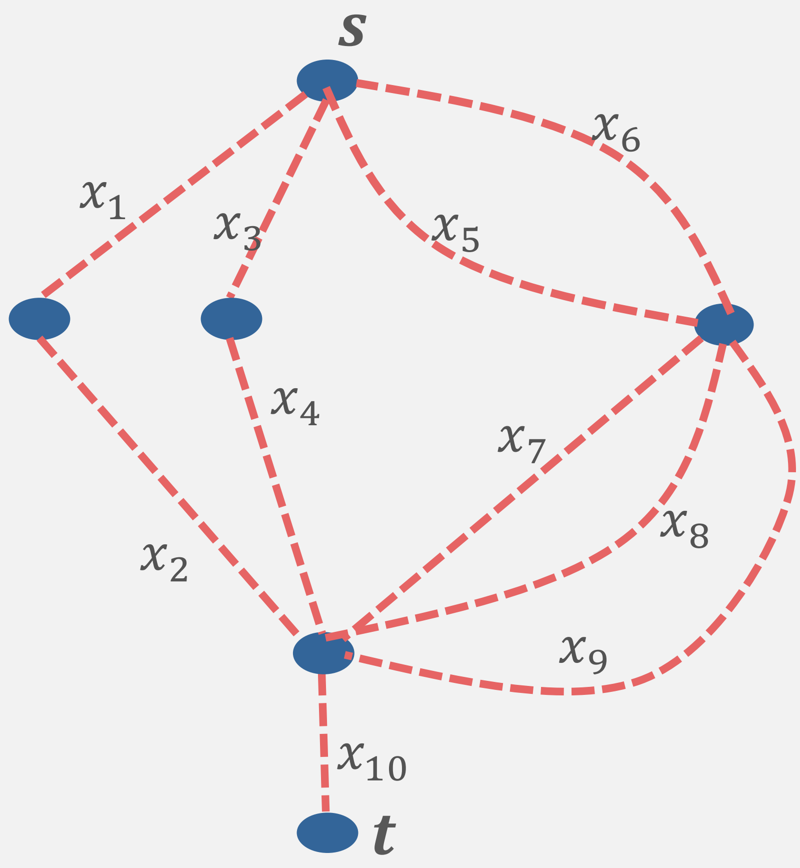
$\vee$ $OR$: outputs 1 if any input subformulas have value 1




$s$ and $t$ are connected if any subgraph is connected

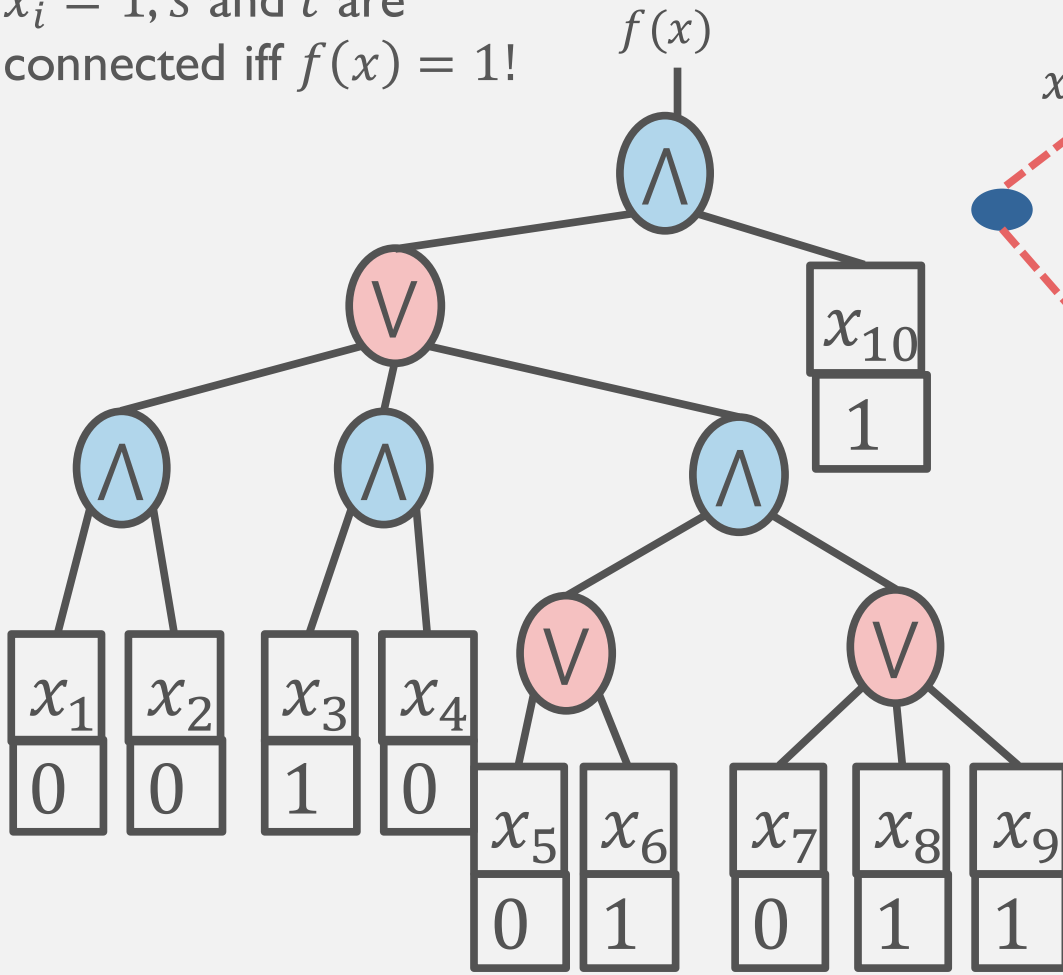
# Application to Boolean Formulas

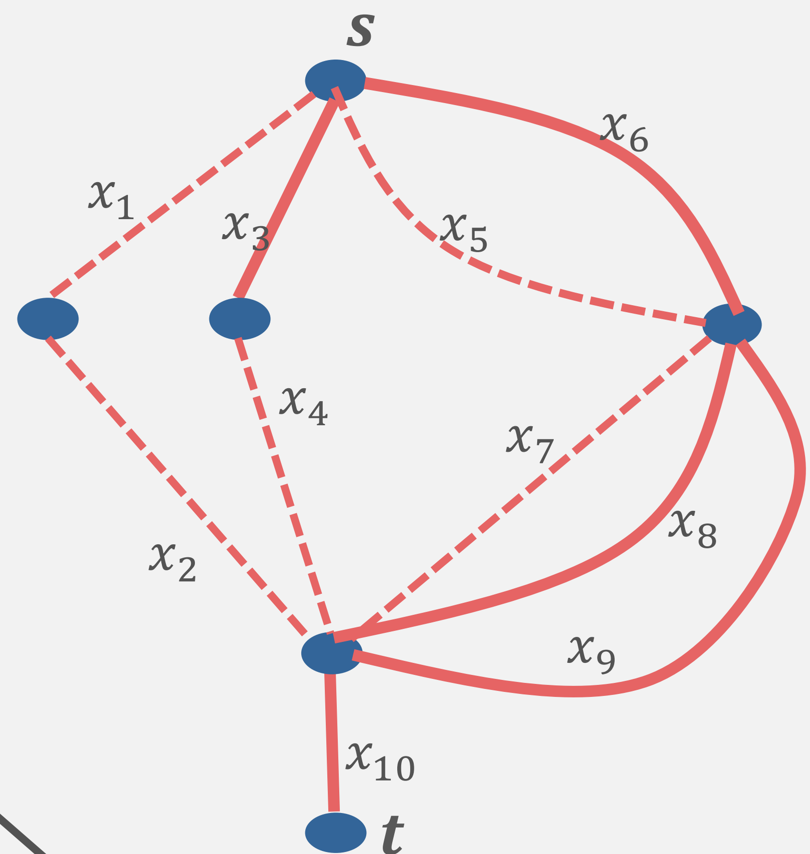


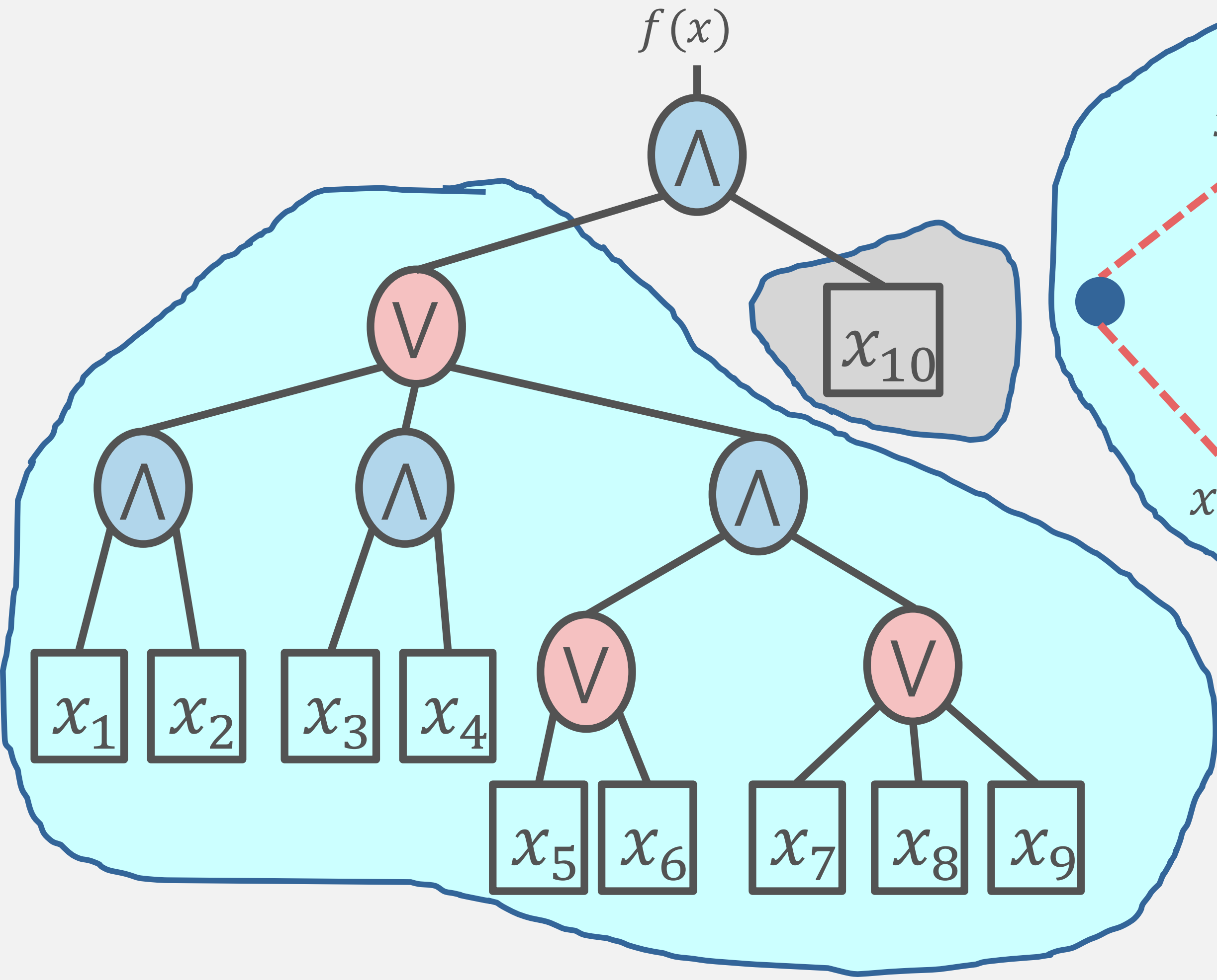
# Application to Boolean Formulas

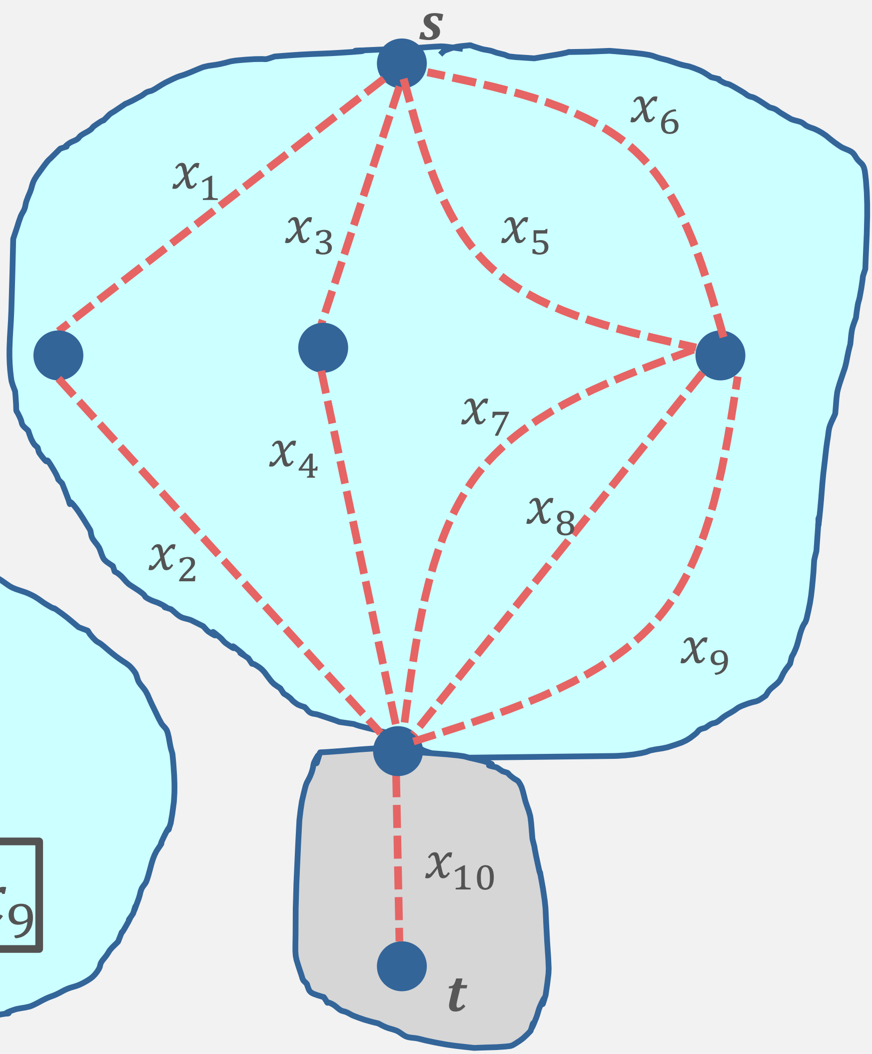
- If we put edges where $x_i = 1$, $s$ and $t$ are connected iff $f(x) = 1$!



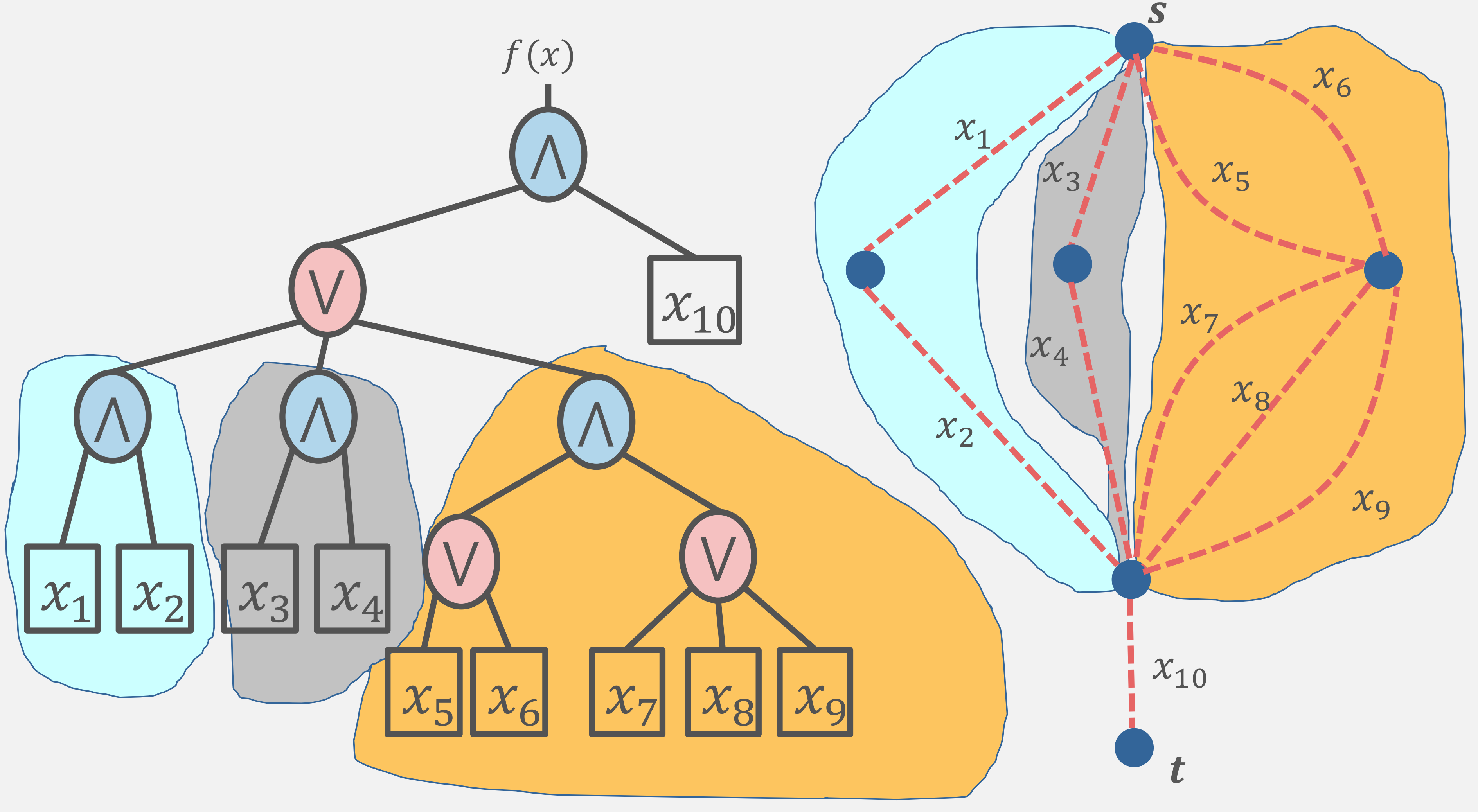



# Application to Boolean Formulas



# Application to Boolean Formulas

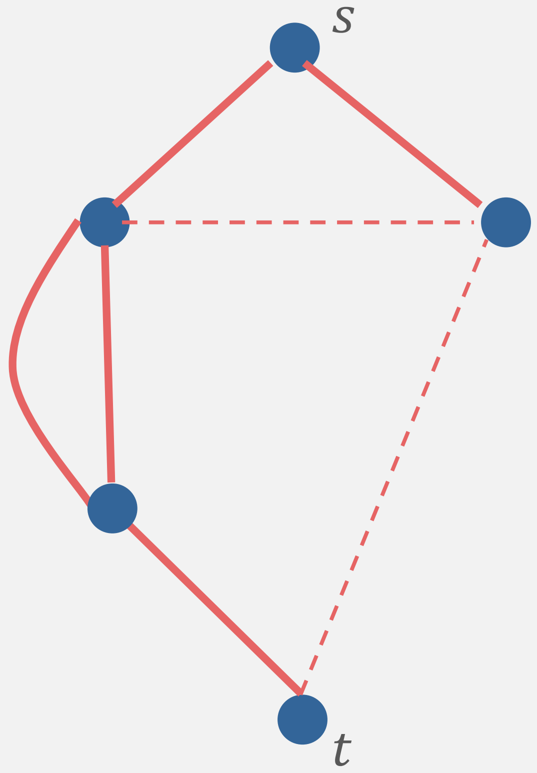


# Outline:

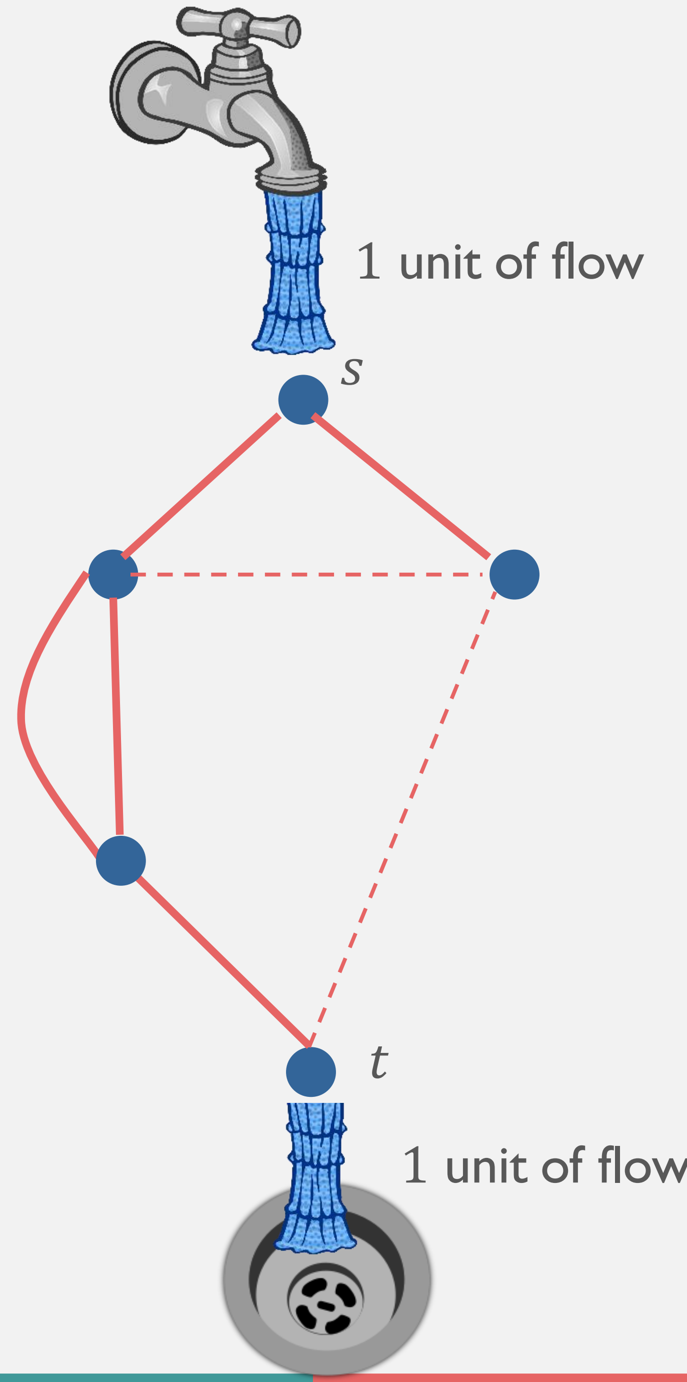
A. Introduction to Quantum Algorithms and st-connectivity
B. st-connectivity makes a good algorithmic primitive
    1. Applies to a wide range of problems
        - Evaluating Boolean formulas reduces to st-connectivity
    2. Easy to understand (without knowing quantum mechanics)

# Effective Resistance

Graph $G$:



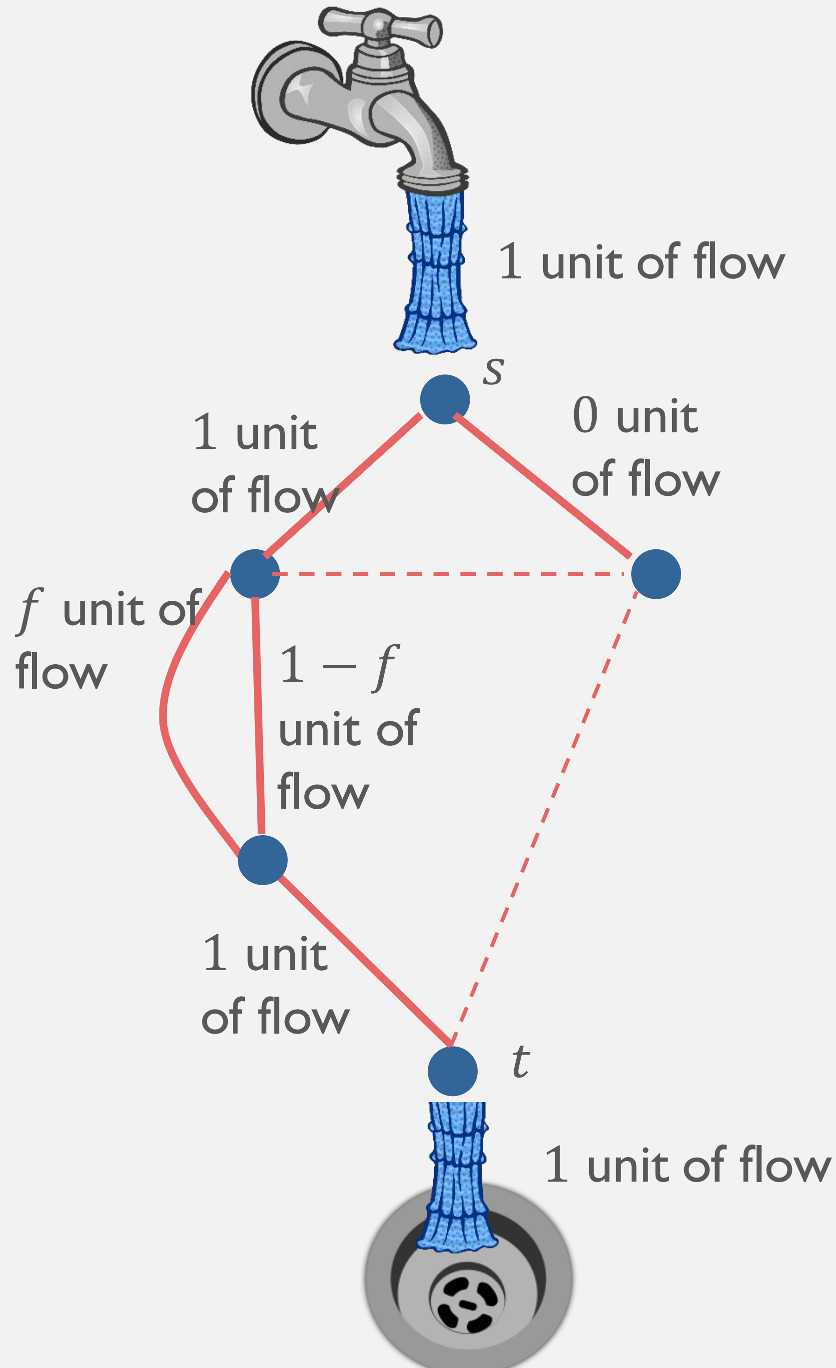
# Effective Resistance



# Effective Resistance

Valid flow:

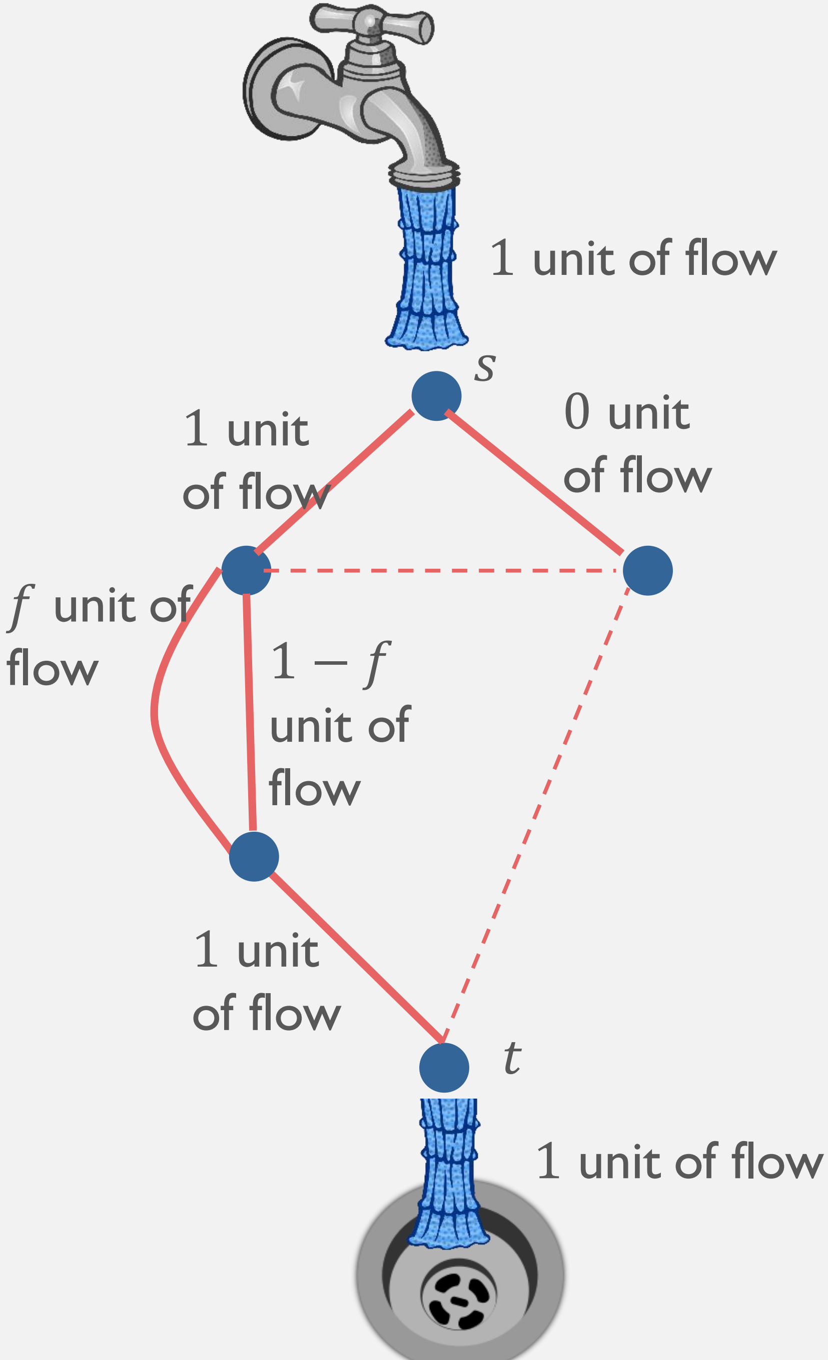
- 1 unit in at $s$
- 1 unit out at $t$
- At all other nodes, zero net flow



# Effective Resistance

Flow energy:

$$\sum_{edges} (flow\ on\ edge)^2$$

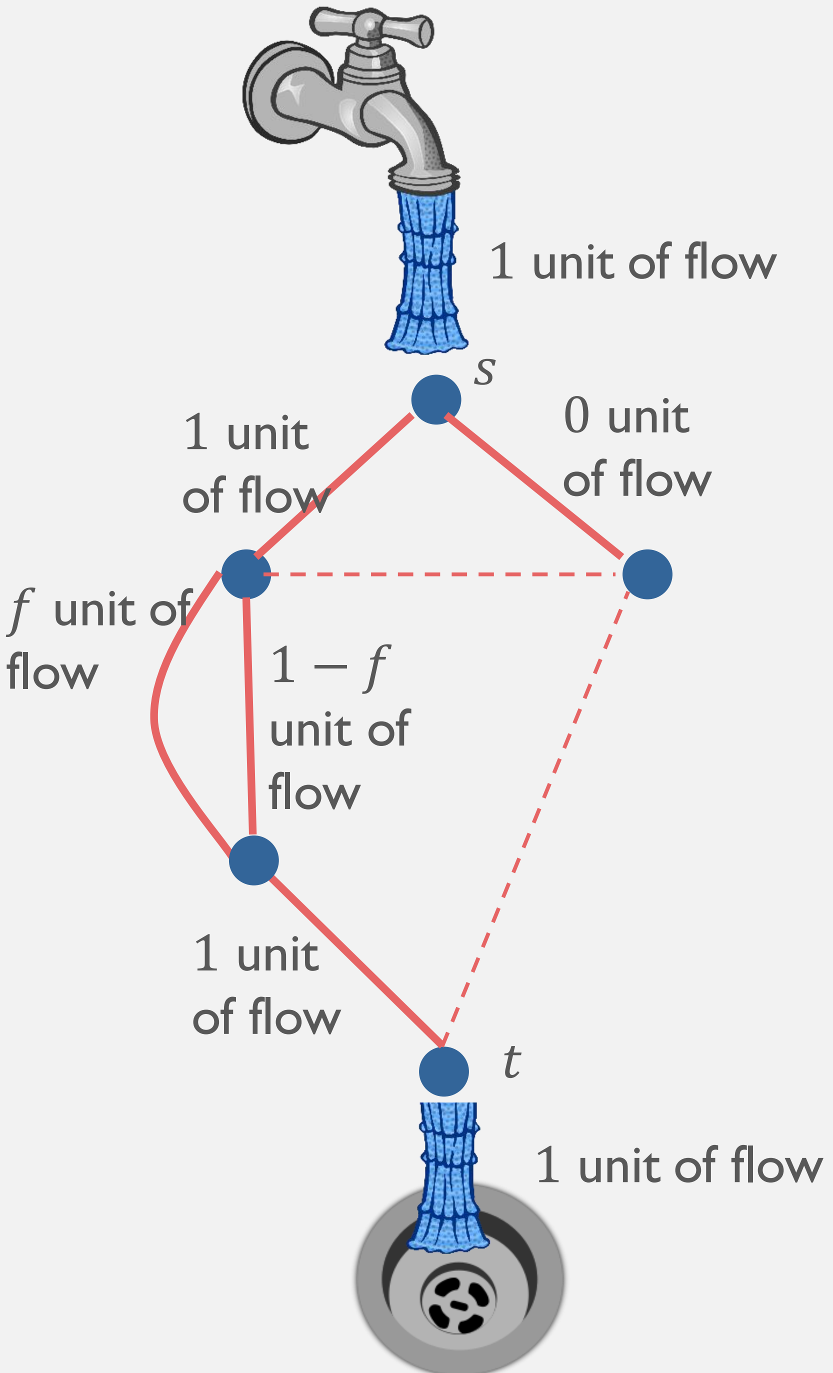


# Effective Resistance

Flow energy:

$$\sum_{edges} (flow\ on\ edge)^2$$

Effective Resistance: $R_{s,t}(G)$
Smallest energy of any valid flow from $s$ to $t$ on $G$.



# Effective Resistance

Flow energy:

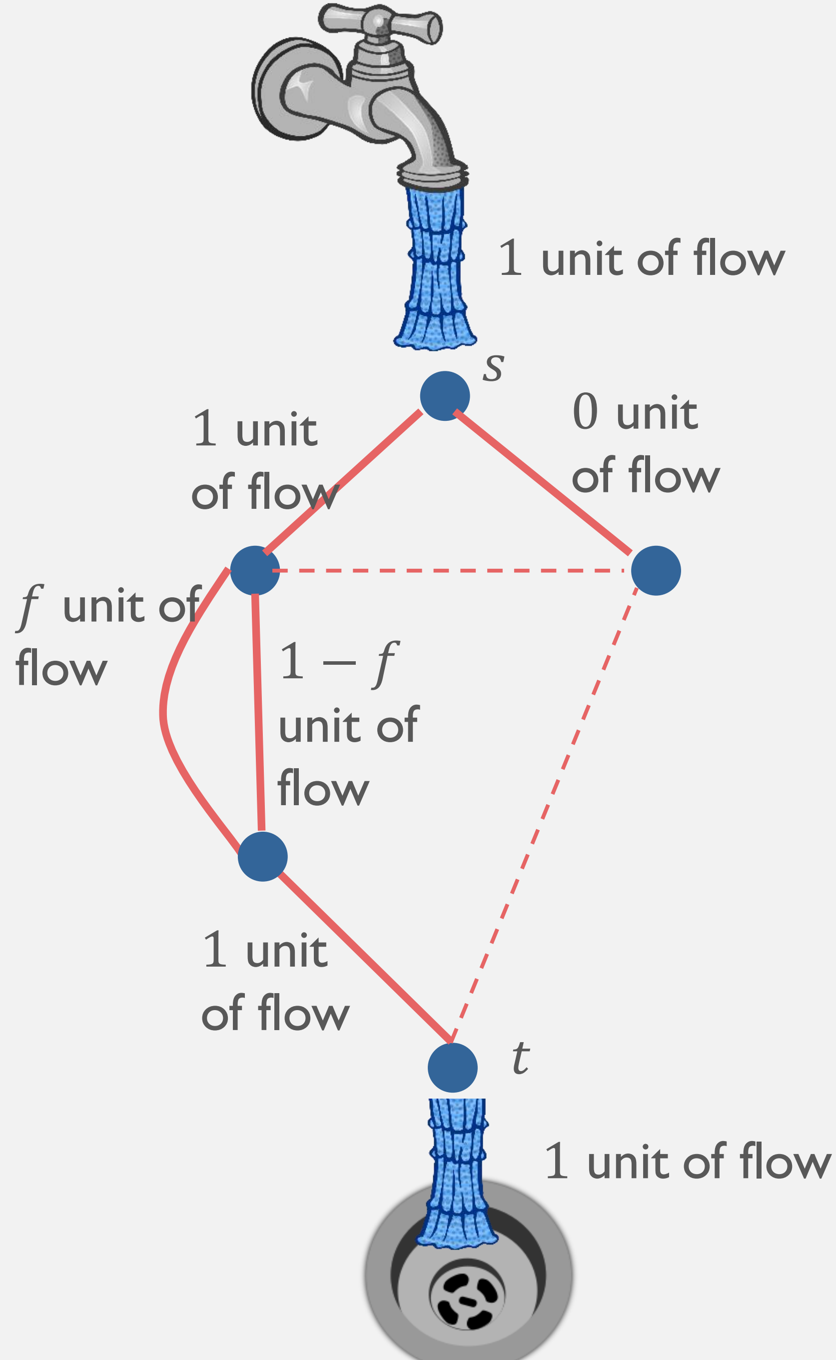
$$\sum_{edges} (flow\ on\ edge)^2$$

Effective Resistance: $R_{s,t}(G)$

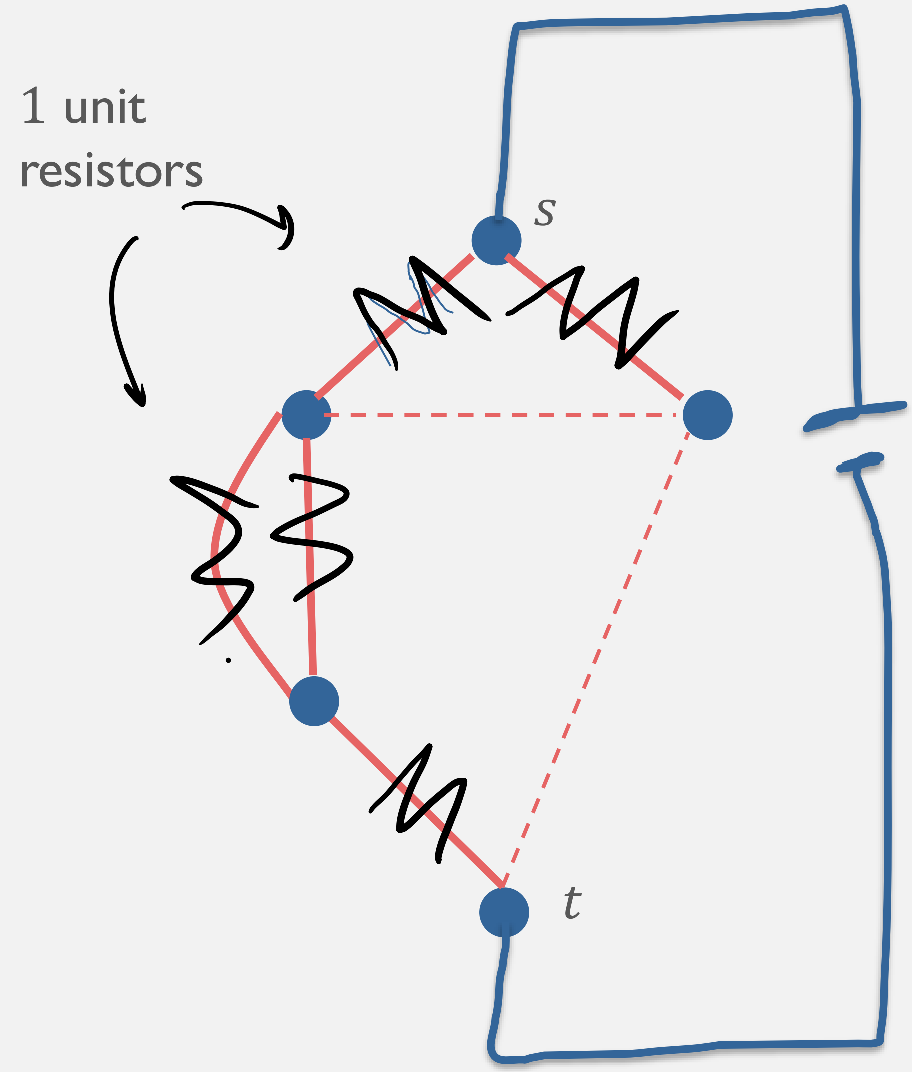
Smallest energy of any valid flow from $s$ to $t$ on $G$.

Properties of $R_{s,t}(G)$

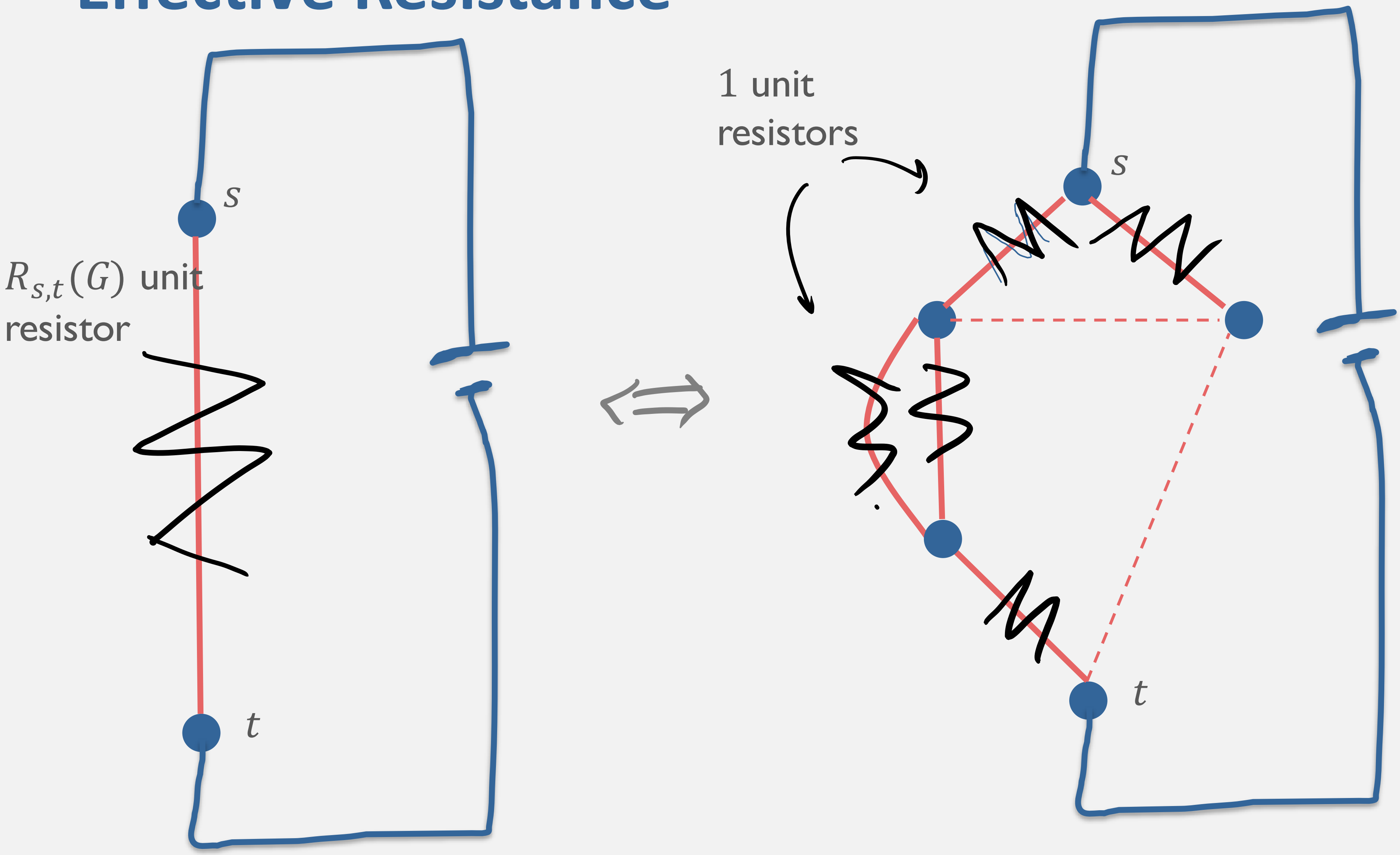
- Small if many short paths from $s$ to $t$
- Large if few long paths from $s$ to $t$
- Infinite if $s$ and $t$ not connected



# Effective Resistance

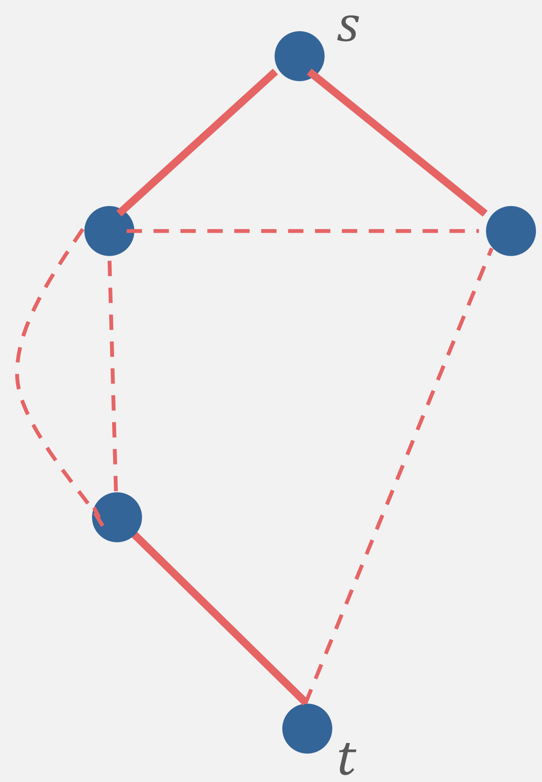


# Effective Resistance



# Effective Capacitance

Graph $G'$:

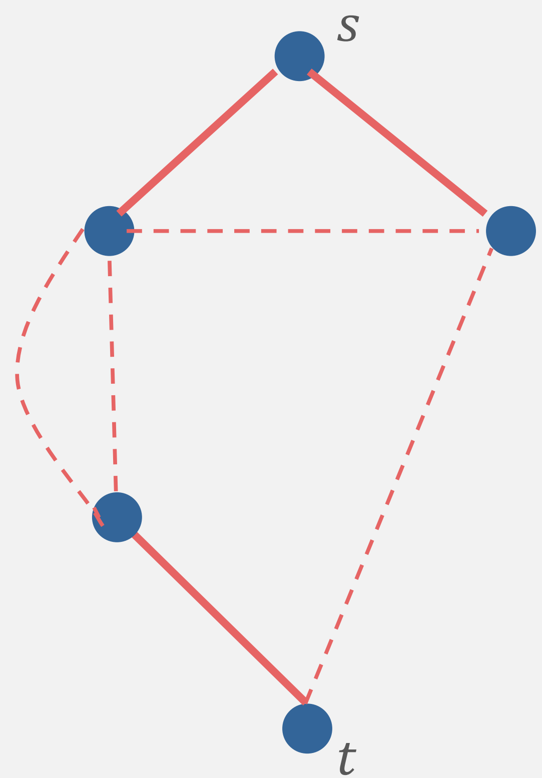


# Effective Capacitance

Valid potential energy:

- $1$ at $s$
- $0$ at $t$
- Potential energy difference is $0$ across edge

Graph $G'$:

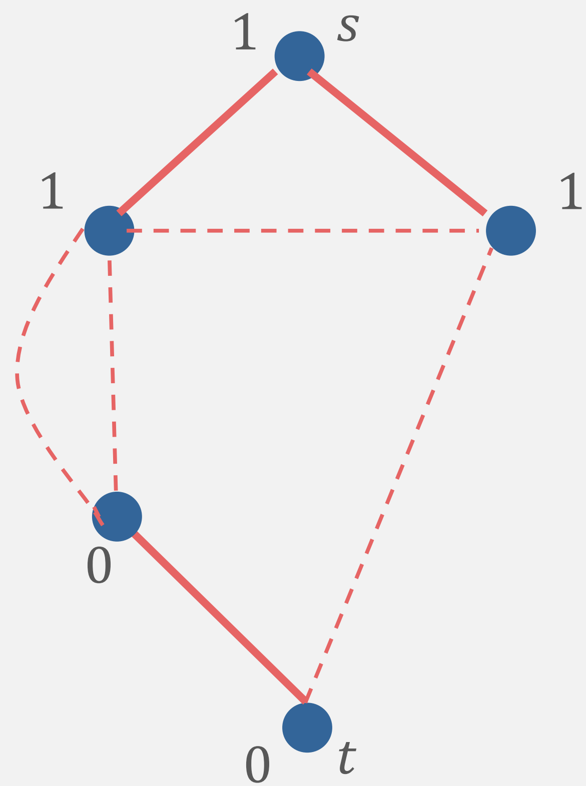


# Effective Capacitance

Valid potential energy:

- $1$ at $s$
- $0$ at $t$
- Potential energy difference is $0$ across edge

Graph $G'$:



# Effective Capacitance

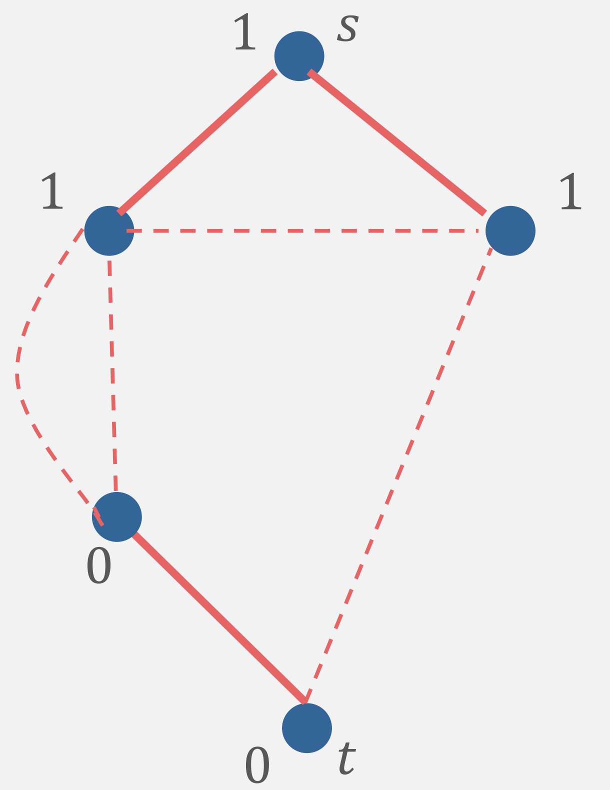
Cut energy:

$$\sum_{edges} (Potential\ Energy\ Difference)^2$$

Effective Capacitance: $C_{s,t}(G')$

Smallest cut energy of any valid potential energy between $s$ to $t$ on $G'$.

Graph $G'$:



# Effective Capacitance

Cut energy:

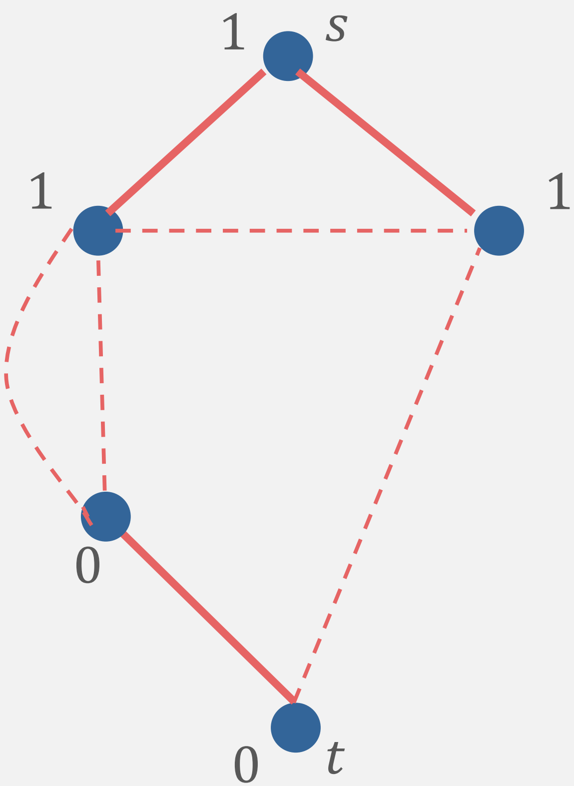
$$\sum_{edges} (Potential\ Energy\ Difference)^2$$

Effective Capacitance: $C_{s,t}(G')$
Smallest cut energy of any valid potential energy between $s$ to $t$ on $G$'.

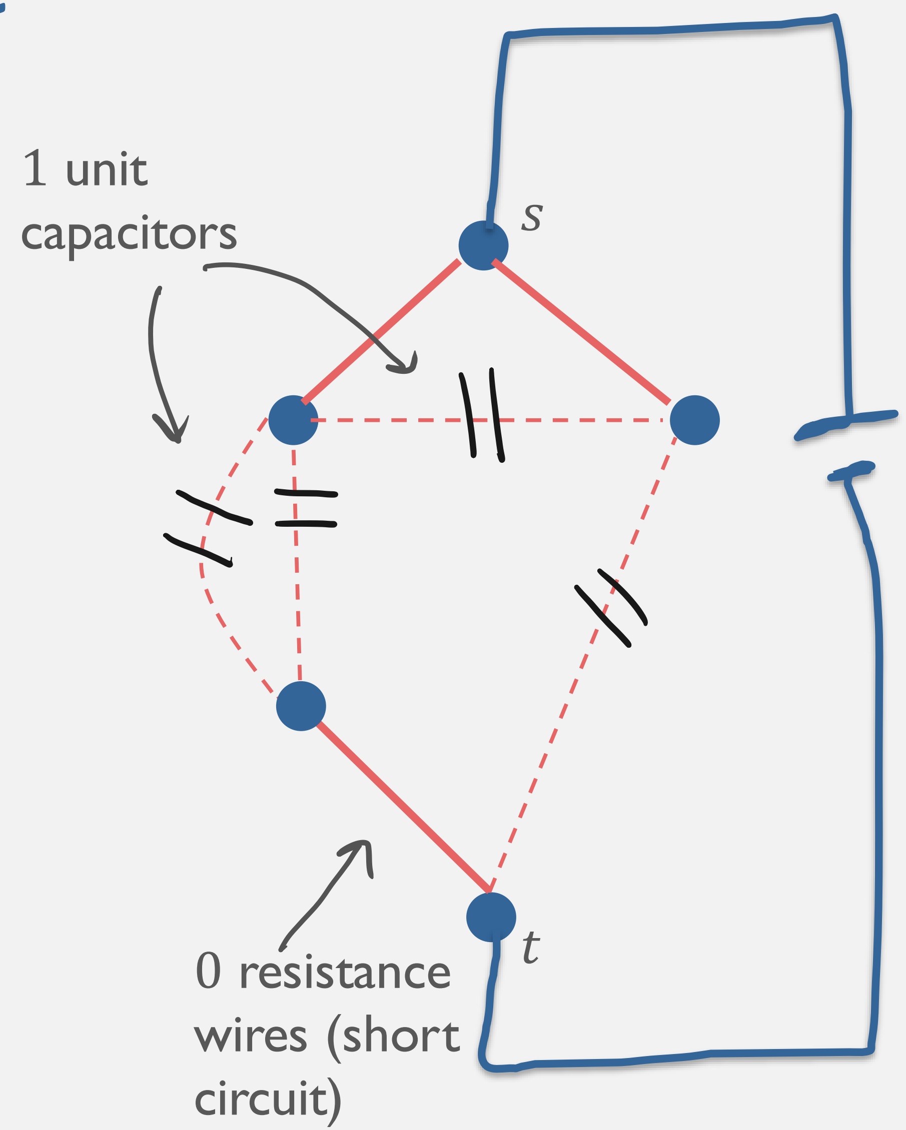
Properties of $C_{s,t}(G')$

- Small if many small cuts
- Large if one large cuts
- Infinite if $s$ and $t$ connected

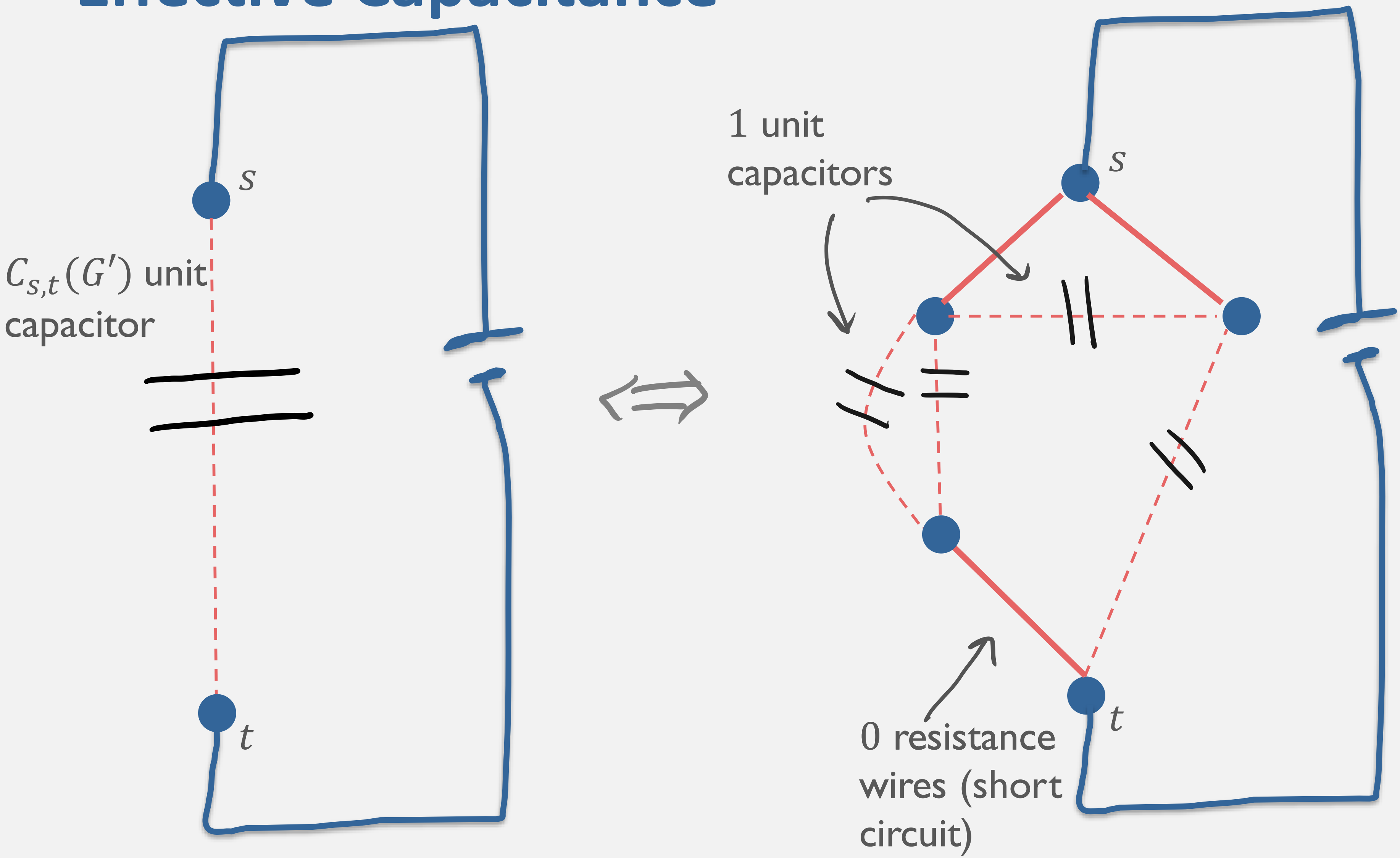
Graph $G'$:



# Effective Capacitance



# Effective Capacitance



# Algorithm Performance:

st-connectivity algorithm complexity =

$$O\left(\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)}\sqrt{\max_{G'\in\mathcal{H}:not\ connected} C_{s,t}(G')}\right)$$

† with $(s,t)$ added also planar

# Algorithm Performance:

st-connectivity algorithm complexity =

$$O\left(\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)}\sqrt{\max_{G'\in\mathcal{H}:not\ connected} C_{s,t}(G')}\right)$$

[Belovs, Reichard, '12] [JJ**K**P, in progress]

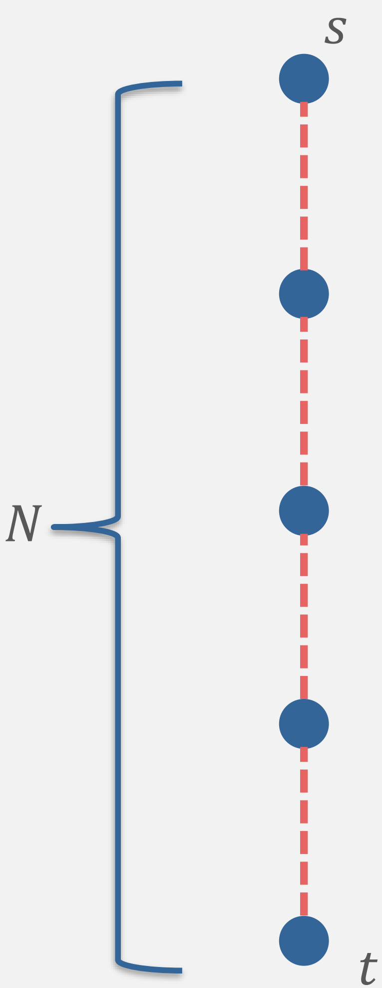
† with $(s,t)$ added also planar

# Example

What is quantum complexity of deciding $AND(x_1, x_2, \ldots, x_N)$, promised

- All $x_i = 1$, or
- At least $\sqrt{N}$ input variables are 0.

# Example



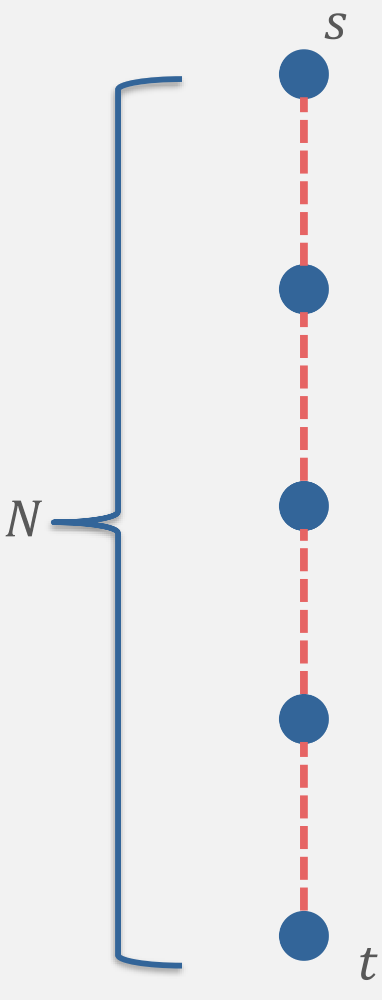

What is quantum complexity of deciding $AND(x_1, x_2, \ldots, x_N)$, promised

- All $x_i = 1$, or
- At least $\sqrt{N}$ input variables are 0.

What is quantum complexity of deciding if

- $s$ and $t$ are connected, or
- At least $\sqrt{N}$ edges are missing

# Example

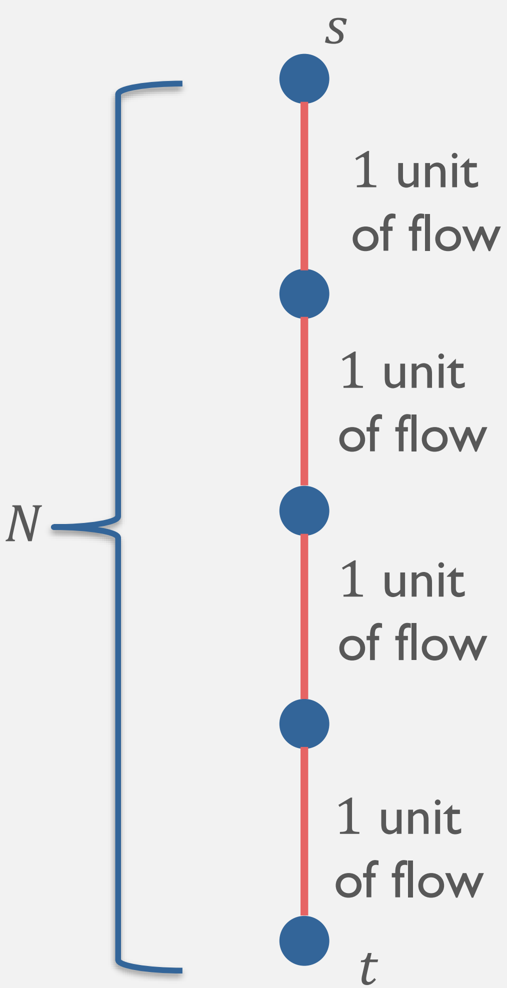



What is quantum complexity of deciding if

- $s$ and $t$ are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)}\ \sqrt{\max_{G'\in\mathcal{H}:not\ connected} C_{s,t}(G')}$$

# Example



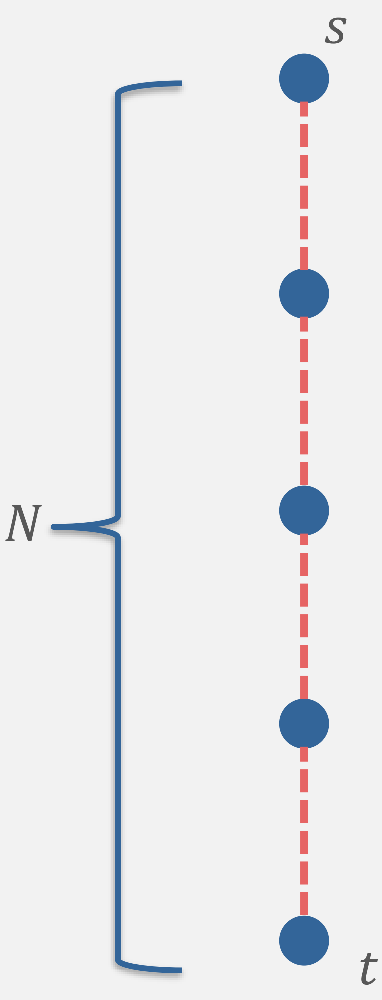

What is quantum complexity of deciding if

- $s$ and $t$ are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)}\ \sqrt{\max_{G'\in\mathcal{H}:not\ connected} C_{s,t}(G')}$$

$$\max_{G\in\mathcal{H}:connected} R_{s,t}(G) = N$$

# Example

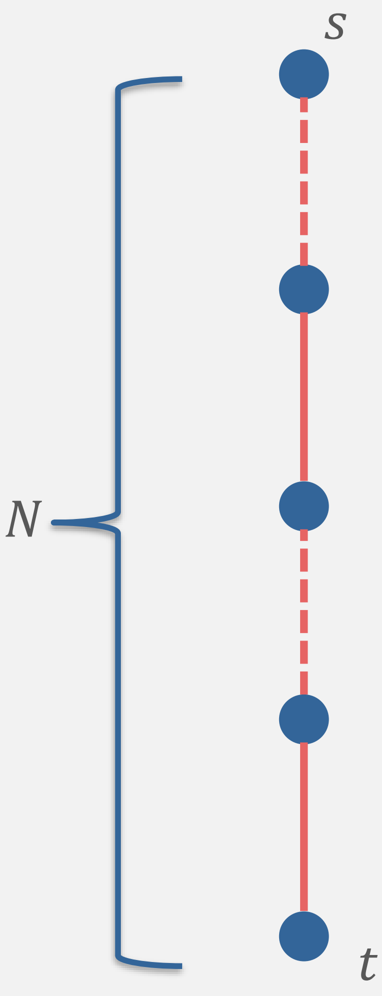



What is quantum complexity of deciding if

- $s$ and $t$ are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G \in \mathcal{H}:connected} R_{s,t}(G)} \sqrt{\max_{G' \in \mathcal{H}:not\ connected} C_{s,t}(G')}$$

# Example

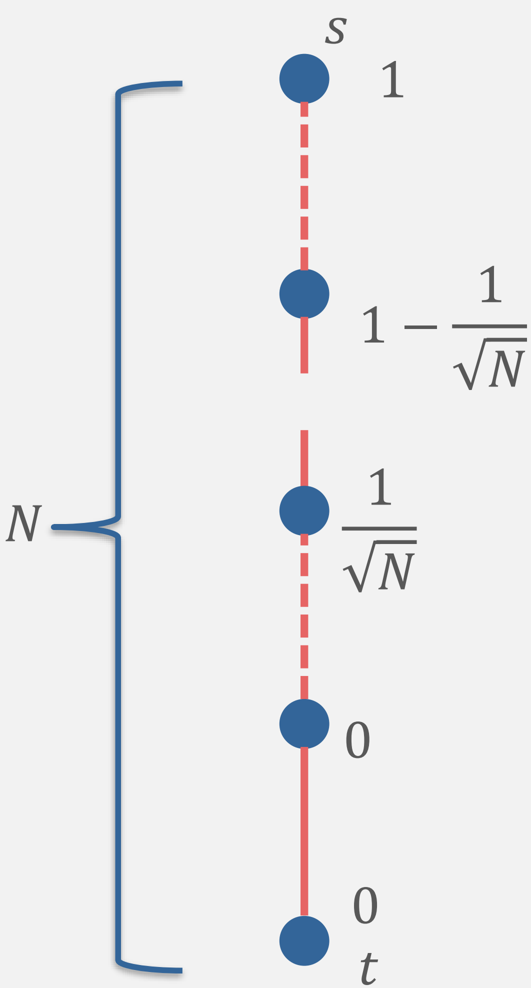



What is quantum complexity of deciding if

- $s$ and $t$ are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)}\ \sqrt{\max_{G'\in\mathcal{H}:not\ connected} C_{s,t}(G')}$$

# Example

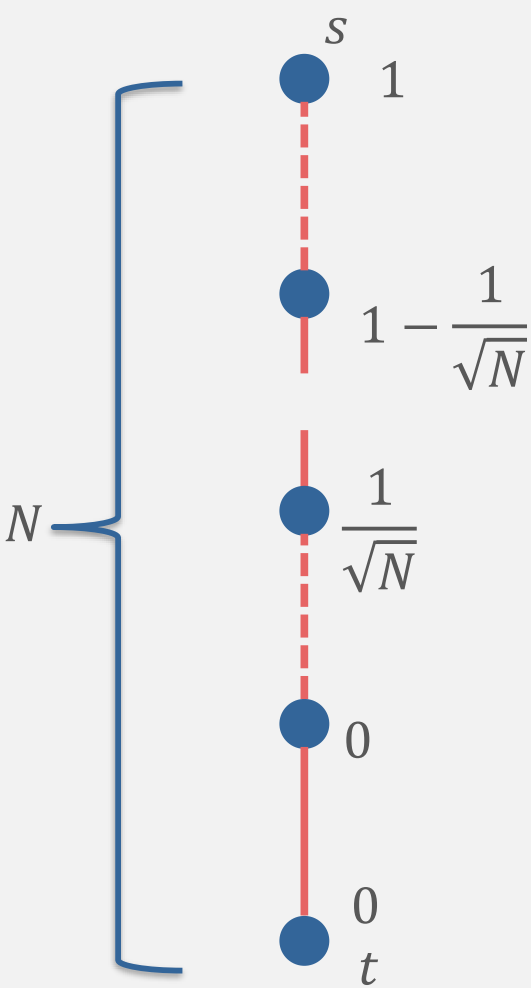



What is quantum complexity of deciding if

- $s$ and $t$ are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G\in\mathcal{H}:connected} R_{s,t}(G)}\ \sqrt{\max_{G'\in\mathcal{H}:not\ connected} C_{s,t}(G')}$$

# Example



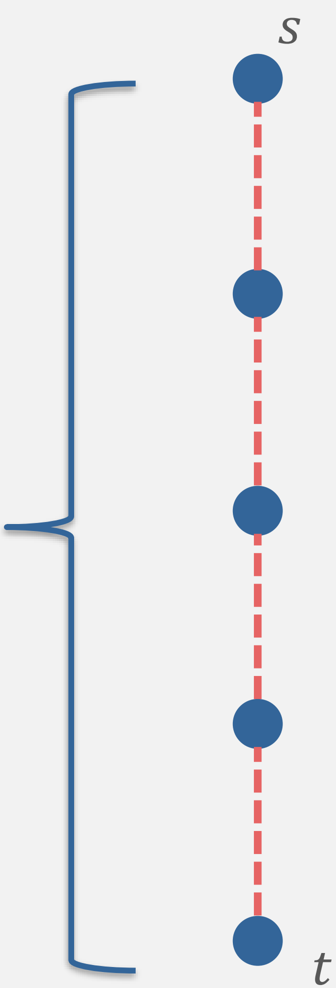

What is quantum complexity of deciding if

- $s$ and $t$ are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G \in \mathcal{H}:connected} R_{s,t}(G)} \sqrt{\max_{G' \in \mathcal{H}:not\ connected} C_{s,t}(G')}$$

$$\max_{G' \in \mathcal{H}:not\ connected} C_{s,t}(G') = \sqrt{N} \times \left(\frac{1}{\sqrt{N}}\right)^2 = \frac{1}{\sqrt{N}}$$

# Example




What is quantum complexity of deciding if

- $s$ and $t$ are connected, or
- At least $\sqrt{N}$ edges are missing

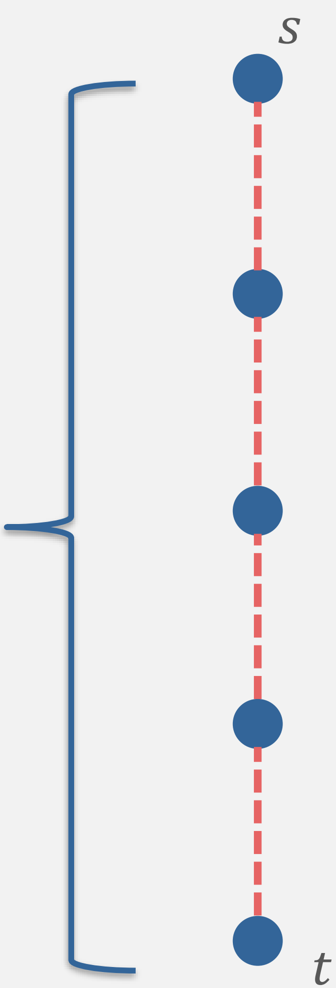
$$\sqrt{\max_{G \in \mathcal{H}:connected} R_{s,t}(G)} \quad \sqrt{\max_{G \in \mathcal{H}:not\ connected} R_{s',t'}(G')}$$

$$\Downarrow \qquad\qquad \Downarrow$$

$$N \qquad\qquad 1/\sqrt{N}$$

Quantum complexity is $O\left(N^{1/4}\right)$

# Example




What is quantum complexity of deciding if

- $s$ and $t$ are connected, or
- At least $\sqrt{N}$ edges are missing

$$\sqrt{\max_{G \in \mathcal{H}:connected} R_{s,t}(G)} \quad \sqrt{\max_{G \in \mathcal{H}:not\ connected} R_{s',t'}(G')}$$

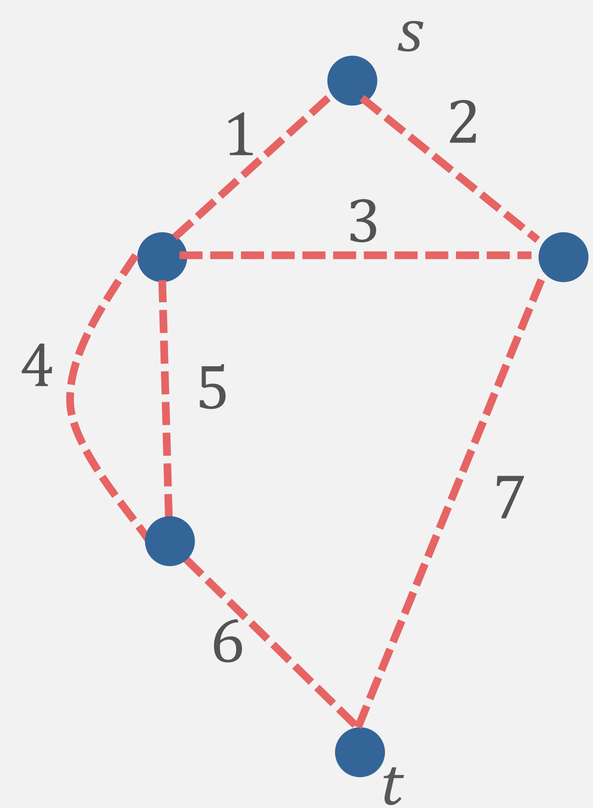
$$N \qquad\qquad 1/\sqrt{N}$$

Quantum complexity is $O\left(N^{1/4}\right)$

Randomized classical complexity is $\Omega\left(N^{1/2}\right)$

# New Example

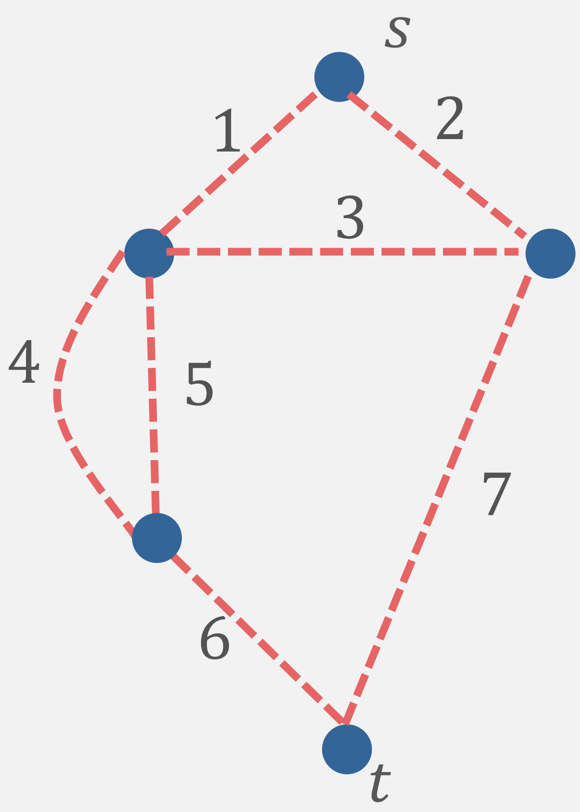
Connectivity – is every vertex connected to every other vertex?



# New Example

Connectivity – is every vertex connected to every other vertex?

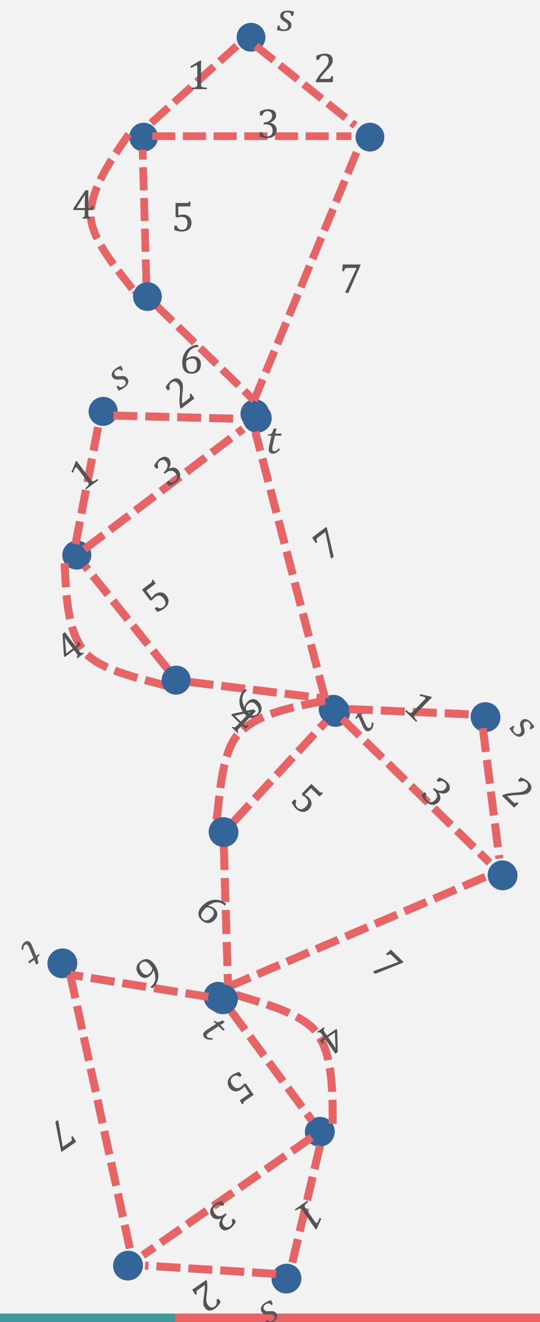
Connectivity=
$(st - conn) \wedge (su - conn) \wedge (uv - conn) \ldots$



# New Example

Connectivity – is every vertex connected to every other vertex?

Connectivity=
$(st - conn) \wedge (su - conn) \wedge (uv - conn) \ldots$

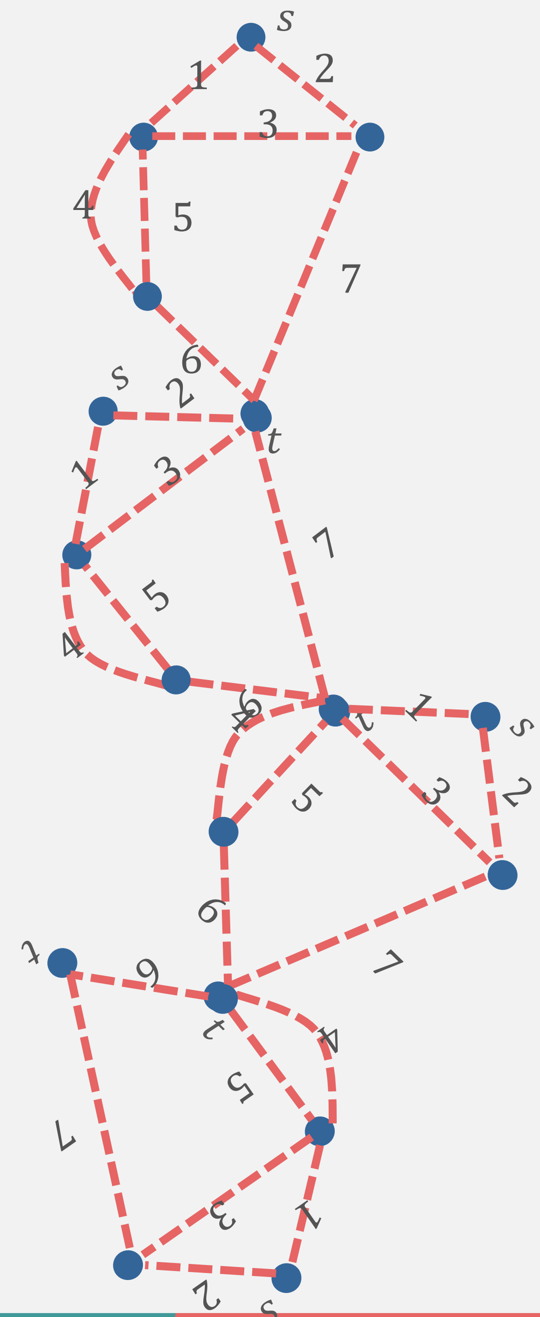


# New Example

Connectivity – is every vertex connected to every other vertex?

Results:

- Worst case: $O(n^{3/2})$ ($n$ = # vertices)
- Promised
  - YES – diameter is $D$
  - NO – every connected component has at most $n^*$ vertices
  - $O\left(\sqrt{nn^*D}\right)$



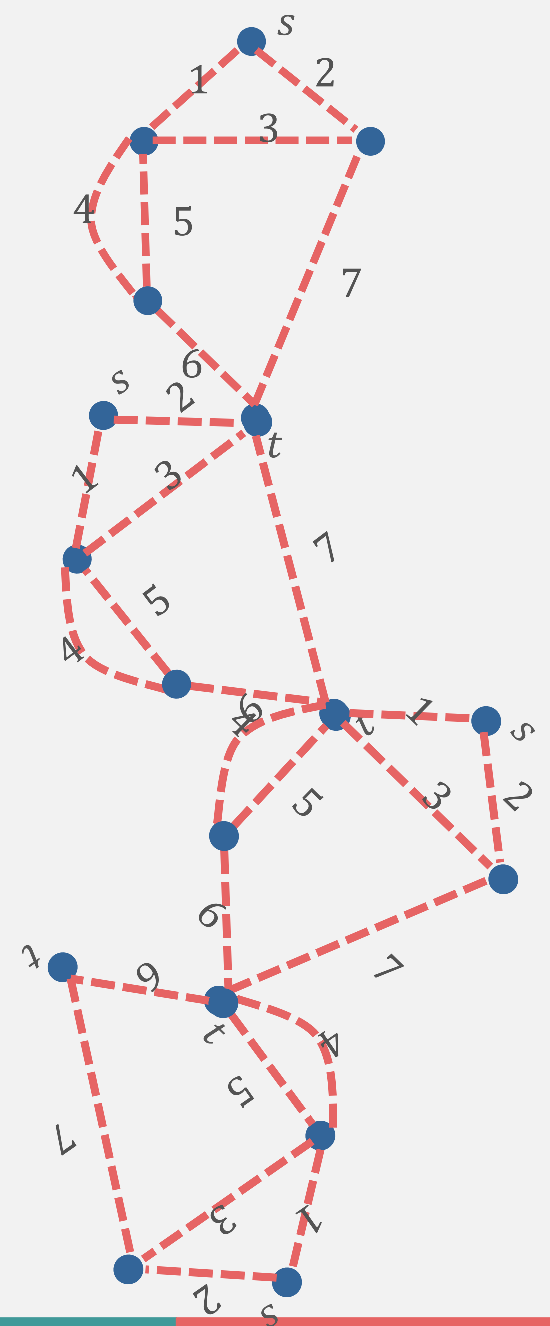
# New Example

Connectivity – is every vertex connected to every other vertex?

Results:

- Worst case: $O(n^{3/2})$ ($n$ = # vertices)
- Promised
  - YES – diameter is $D$
  - NO – every connected component has at most $n^*$ vertices
  - $O\left(\sqrt{nn^*D}\right)$

(Diameter result previously discovered by Arins using slightly different approach)



# The Algorithm

Span Program

- Span vectors
- Target vector

The input to the problem determines which subset of span vectors are allowed.

If target vector is in span of the allowed span vectors, then function evaluates to $1$ on that input. Otherwise, evaluates to $0$.

Thus span program encodes a function.

Infinite number of span programs can encode the same function

Given a span program, can create a quantum algorithm to evaluate the corresponding function (create a quantum walk whose dispersion operators are based on the vectors)

# The Algorithm

Span Program

- Span vectors
- Target vector

Given a span program, can create a quantum algorithm to evaluate the corresponding function (create a quantum walk whose dispersion operators are based on the vectors)

The efficiency of the span program is a (relatively) simple function of the vectors.

There is always a span program algorithm that is optimal (and many that are not optimal.)

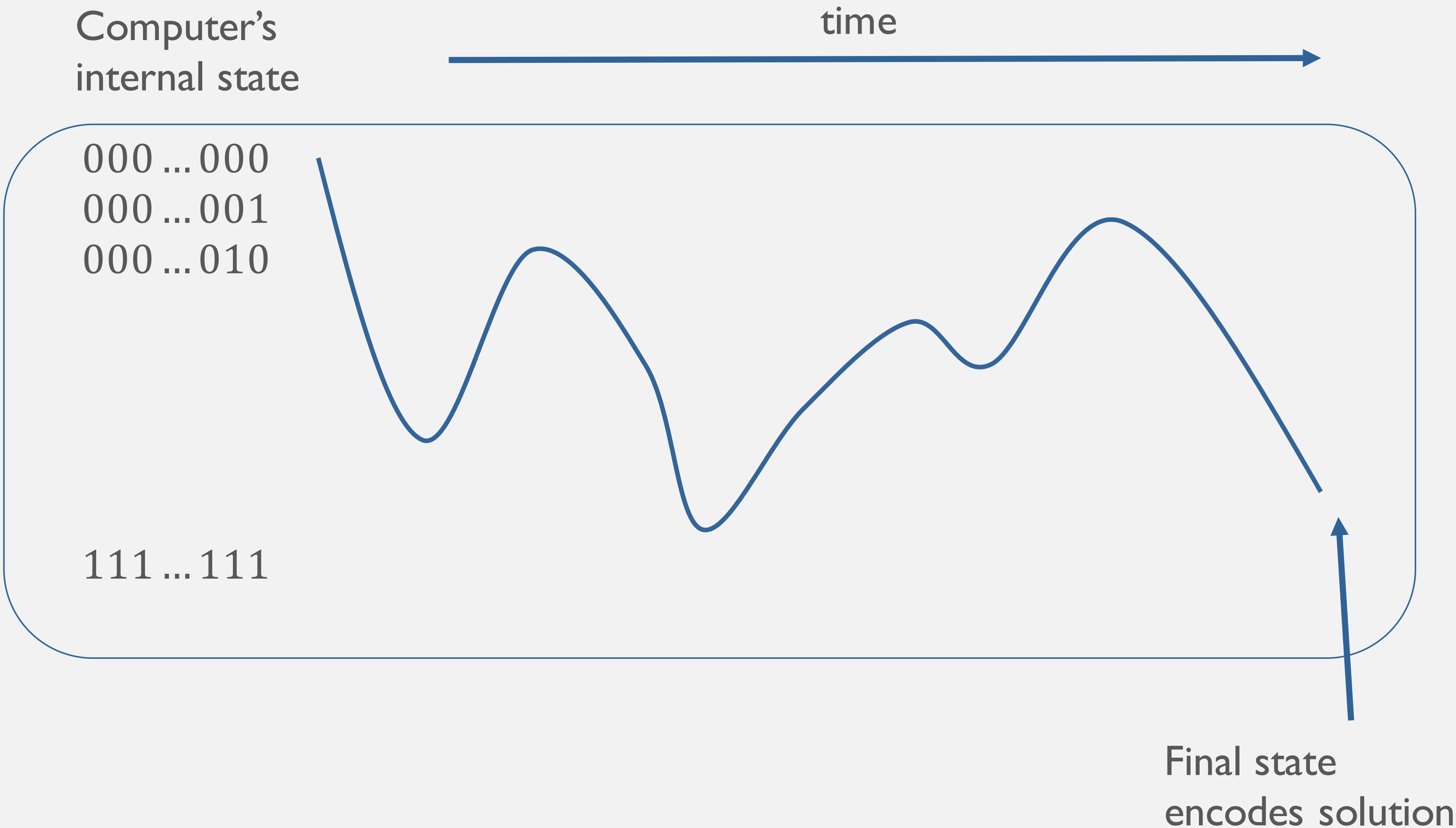
# Open Questions and Current Directions

- When is our algorithm optimal for Boolean formulas? (Especially partial/read-many formulas)
- Are there other problems that reduce to st-connectivity? (Perhaps all?)
- What is the classical time/query complexity of st-connectivity in the black box model? Under the promise of small capacitance/resistance?
- Does our reduction from formulas to connectivity give good classical algorithms too?
- How to choose weights?

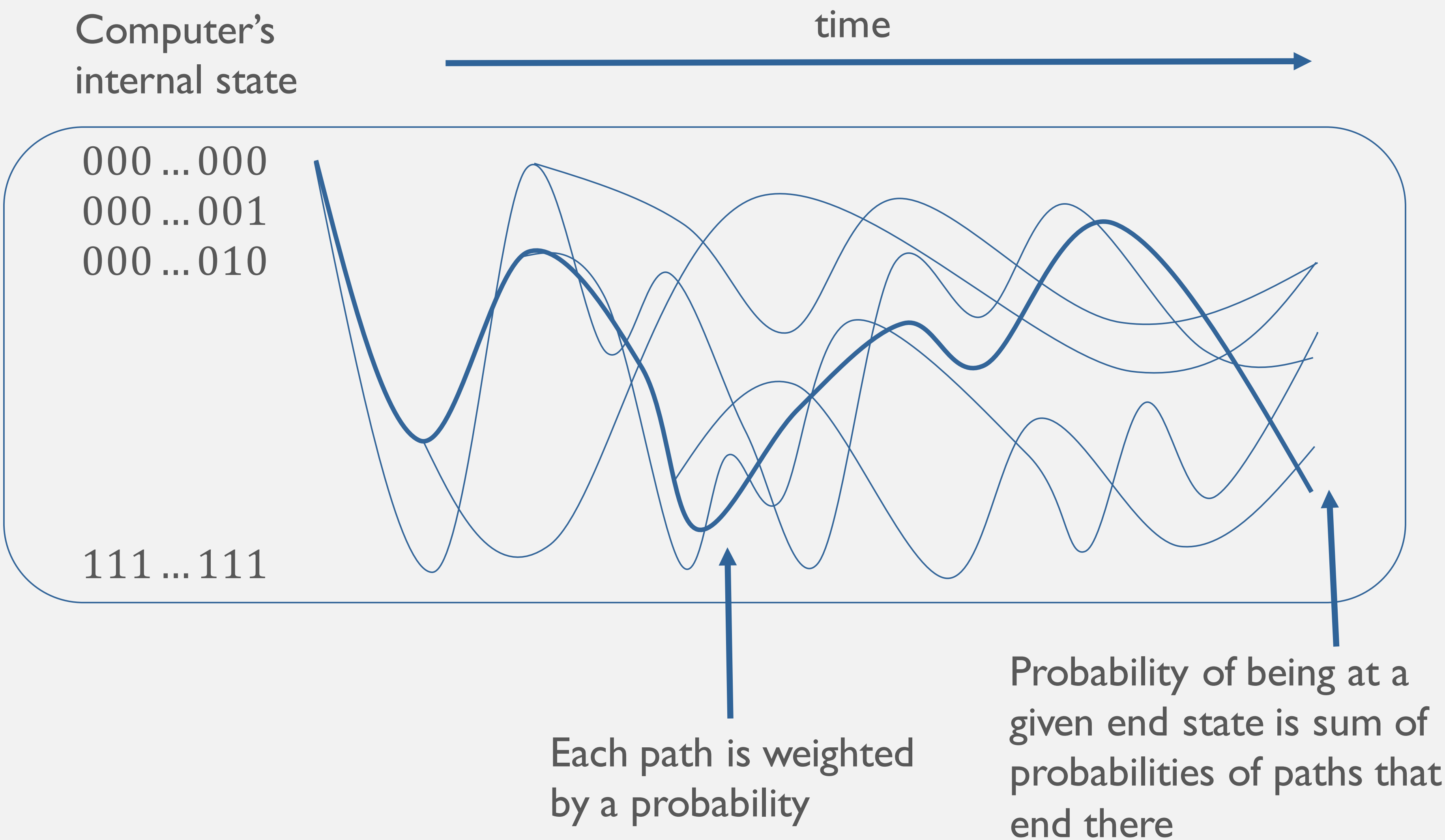
# Other interests

- Statistical inference and machine learning applied to quantum characterization problems
- Quantum complexity theory, especially quantum versions of NP

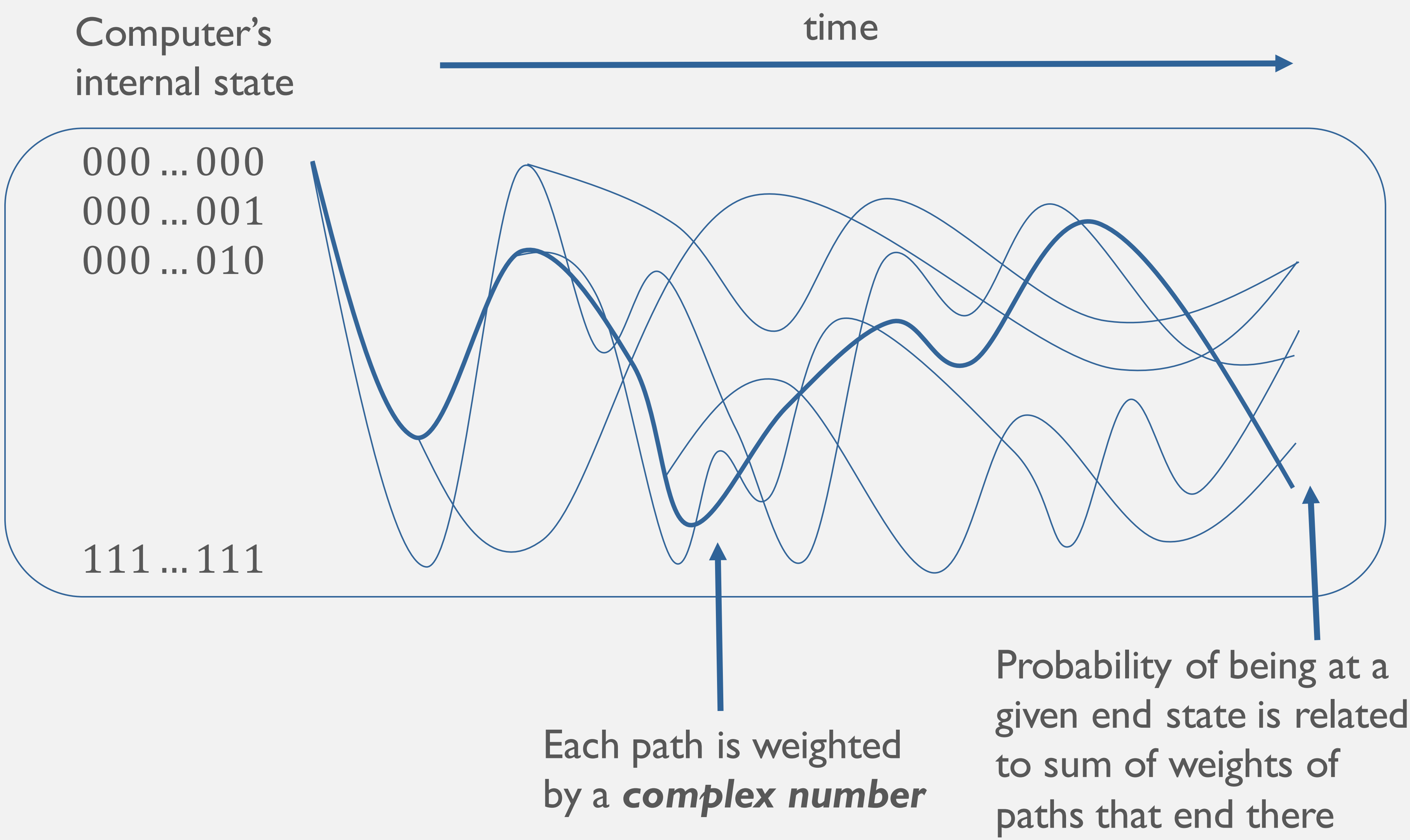
# Classical Computing



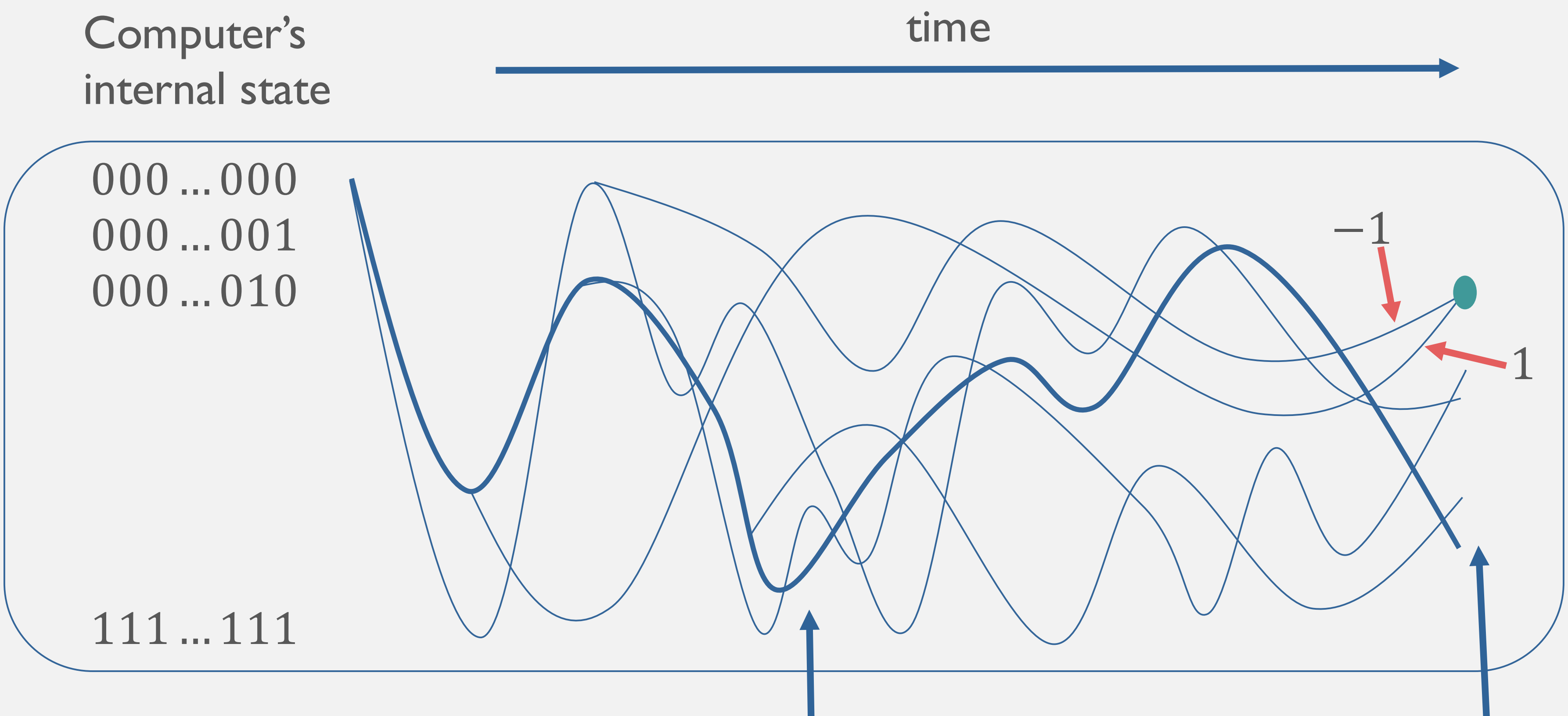
# Probabilistic Computing



# Quantum Computing



# Quantum Computing




Each path is weighted by a ***complex number***

Probability of being at a given end state is related to sum of weights of paths that end there

# Figure of Merit

- **Quantum** Query Complexity
  - Counts number of uses (queries) of the black box **(inputs can be queried in quantum superposition)**
  - All other operations are free
  - Imagine the black box is a hard to compute function, so we want to minimize the number of times we use it.

- **Quantum** Time Complexity
  - Counts the total number of **quantum** operations, including uses of black box.

# Figure of Merit

- **Quantum** Query Complexity
  - Counts number of uses (queries) of the black box **(inputs can be queried in quantum superposition)**
  - All other operations are free
  - Imagine the black box is a hard to compute function, so we want to minimize the number of times we use it.

- **Quantum** Time Complexity
  - Counts the total number of **quantum** operations, including uses of black box.

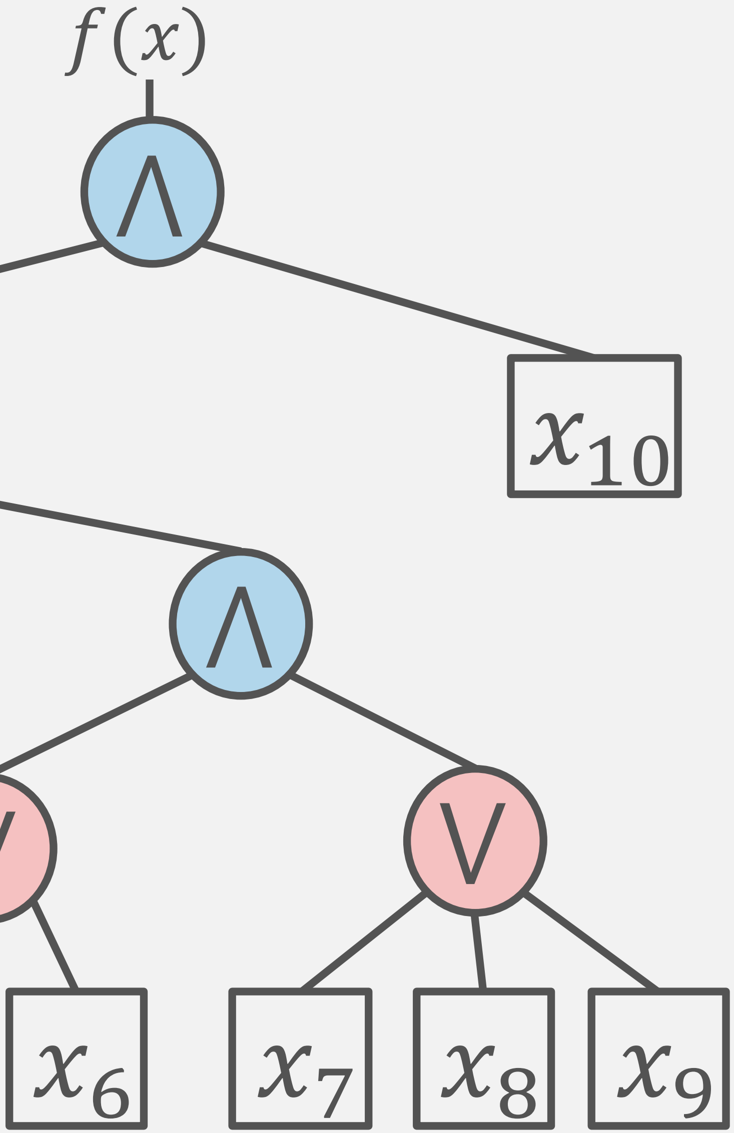
Under a mild assumption, these two will be the same for our algorithm up to a logarithmic factor.

# Boolean Formulas

$\wedge$ $AND$: outputs 1 if all inputs are 1

$\vee$ $OR$: outputs 1 if any input is 1

$x_i$ Value 0 or 1

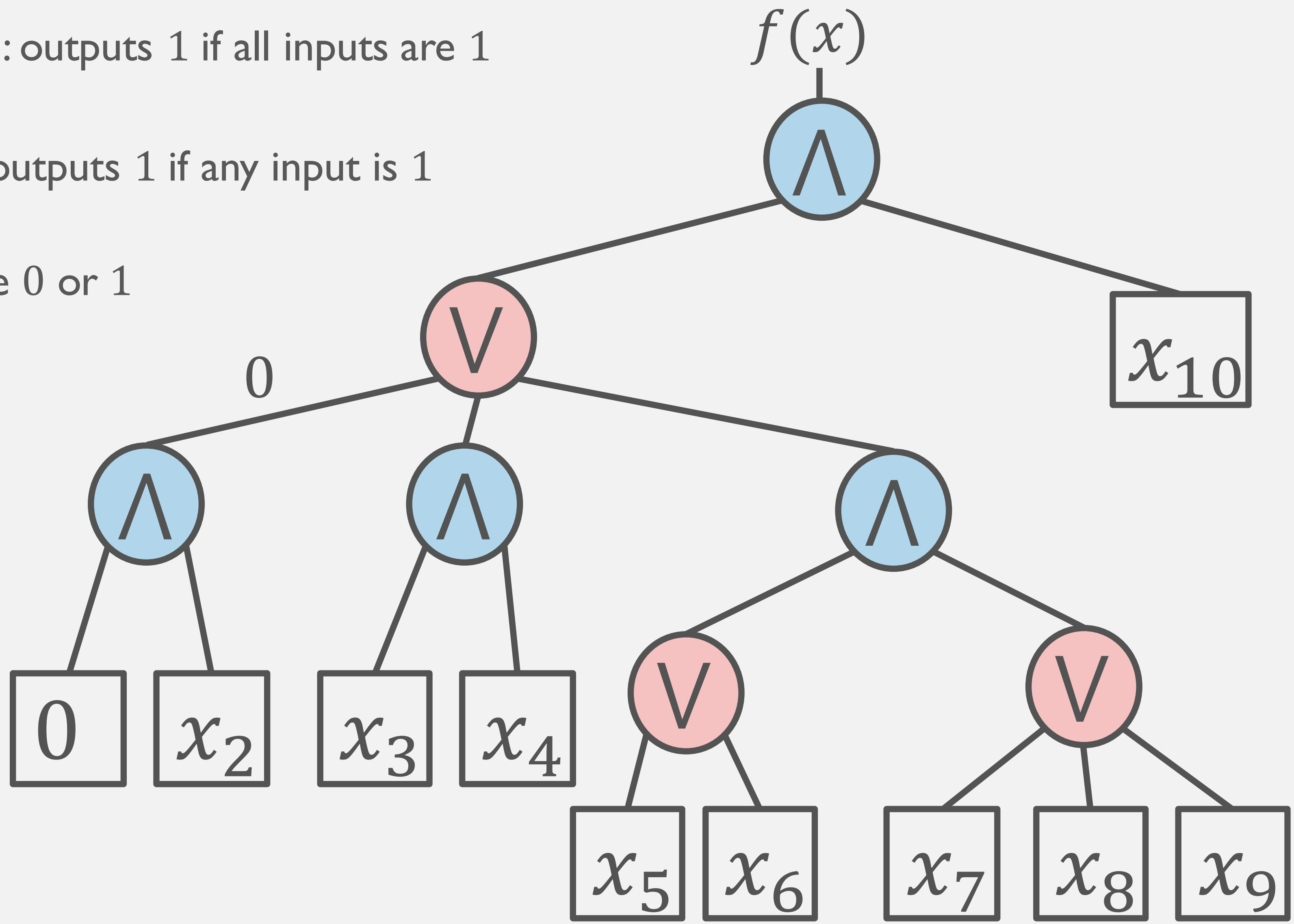


# Boolean Formulas

$\wedge$ $AND$: outputs 1 if all inputs are 1

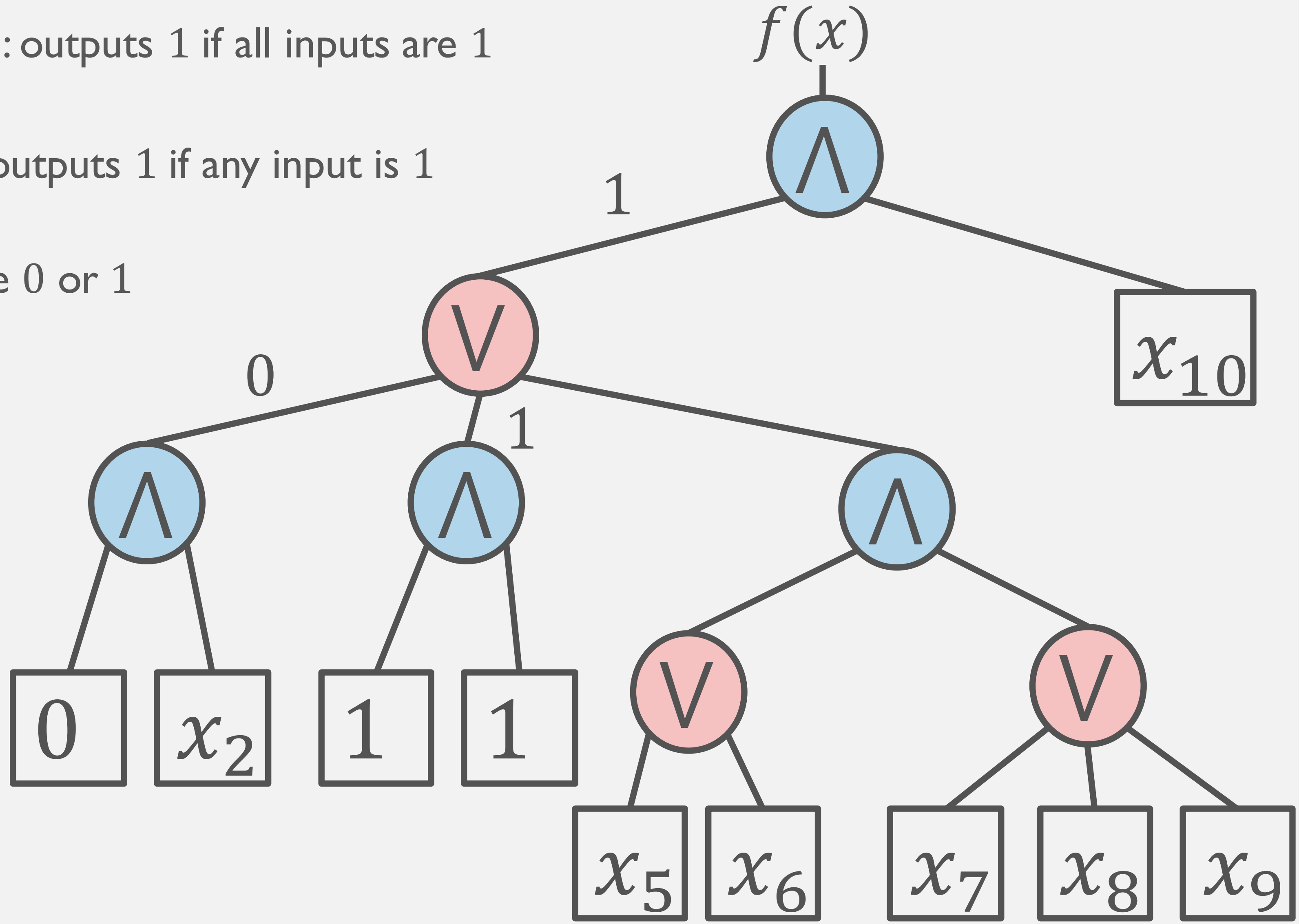
$\vee$ $OR$: outputs 1 if any input is 1

$x_i$ Value 0 or 1



# Boolean Formulas

- $\wedge$ $AND$: outputs 1 if all inputs are 1
- $\vee$ $OR$: outputs 1 if any input is 1
- $x_1$ Value 0 or 1



# Boolean Formulas

$\wedge$ $AND$: outputs 1 if all inputs are 1

$\vee$ $OR$: outputs 1 if any input is 1

$x_1$ Value 0 or 1